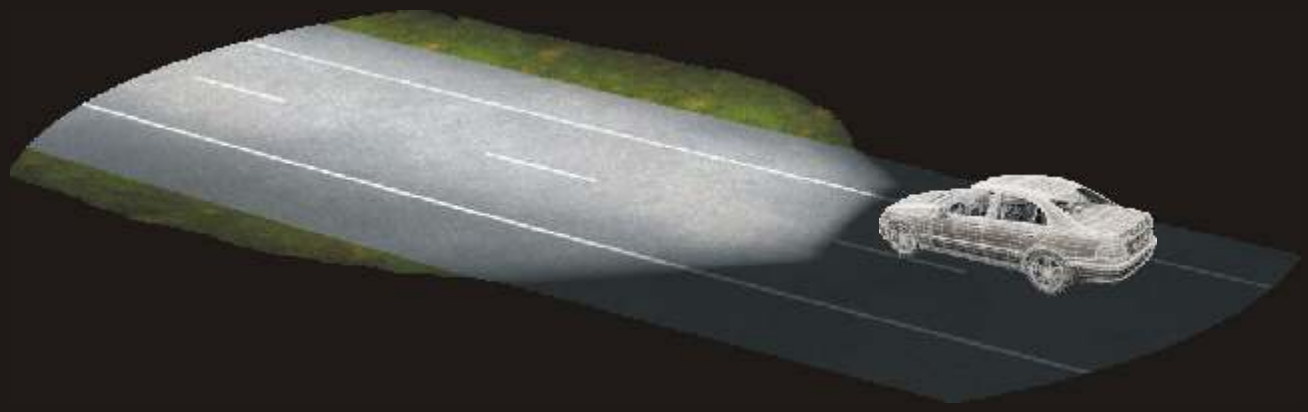
LucidDrive

Night Driving Simulation



User's Manual

brandenburg gmbh
software solutions for the product development

LucidDrive Manual

Version 1.6.0

Published September 12, 2013

Table of Contents

1. Overview 5
    1.1. How to use the Documentation 5
    1.2. What is LucidDrive? 5
    1.3. Road Scenes 5
    1.4. Beam Pattern Analysis 6
    1.5. Advanced Front Lighting Systems (AFS) 6
    1.6. Texture Based Graphics 6
    1.7. The LucidDrive Coordinates Systems 6
2. Getting Started with LucidDrive 9
    2.1. The Setup of LucidDrive 9
        2.1.1. Installation 9
        2.1.2. License File Handling 10
    2.2. Tool Bar Symbols 11
    2.3. Geo View and Message View 12
3. The Menu Items in LucidDrive 13
    3.1. Configurations 13
        3.1.1. Load Config File 14
        3.1.2. Save Config File 14
        3.1.3. Predefined Config Files 14
    3.2. Roads 16
    3.3. Headlamps 17
        3.3.1. Main Dialog 17
        3.3.2. Headlamp Advanced Options and Parameters 19
    3.4. Lamp Switch 20
    3.5. Visual Properties 20
    3.6. Driving 23
    3.7. Scripts/Plug-ins (Vehicle Dynamics, AFS, Aux Plug-ins, Trigger Points) 24
    3.8. Video 24
    3.9. Cameras 25
    3.10. Scene 27
    3.11. Settings 29
    3.12. Miscellaneous Menu Commands 30
4. Multiple Cars 31
    4.1. Other Settings 31
5. Scripts, AFS and Vehicle Dynamics Plug-ins 33
    5.1. Introduction 33
    5.2. LucidDrive Conventions 33
    5.3. Plug-in Classification and Pipeline 33
    5.4. Plug-in Architecture 35
    5.5. Signal-Slot Mechanism 36
6. Plug-in Script Interface 39
    6.1. Lamp functions 39
    6.2. Vehicle functions 41
7. Plug-ins provided in LucidDrive 45
    7.1. AFS Scripts 45
    7.2. Auxiliary Scripts 47
8. Physics on Geometry, BRDF, Monitor Luminance, etc. 49
    8.1. Geometry 49
    8.2. BRDF 50

8.2.1. Hoffmann & Nakata 50
8.2.2. Lommel-Seeliger 51
8.3. Illumination & Luminance Parameters 52
8.4. Monitor and Luminance 53
9. LucidDrive Keyboard Shortcuts 55
9.1. LucidDrive Shortcuts 55
9.2. LucidDriveManual and LucidDriveServerClient Extra Shortcuts 55

Chapter 1. Overview

1.1. How to use the Documentation

This documentation has two purposes: it introduces you to the use of LucidDrive, and it serves as a reference handbook in your daily work.

It is assumed that you know about optics and lighting and its terminology. Unique definitions of the terminology used in this handbook are given in the glossary. It is also assumed that you are acquainted with the standard Microsoft Windows® dialogs like File Open, Save, etc.

The installation of LucidDrive is described in Section 2.1, “The Setup of LucidDrive”. Chapter 3, *The Menu Items in LucidDrive* describes all menu items of LucidDrive in detail.

1.2. What is LucidDrive?

LucidDrive is a tool to evaluate beam patterns and to develop desired headlamp motion in dynamic driving simulations. LucidDrive complements the design tools by allowing an analysis of automotive headlamps with various road scenes. It is a software tool to provide the virtual reality simulation, prior to expensive fabrication and testing. It runs in an automatic driving mode without a need for interactive control like steering, accelerating and breaking. The primary function of LucidDrive is to evaluate the beam pattern under conditions as realistic as possible. One can switch quickly between different lamps, viewing positions, road types and scenes during a simulation, thus receiving an immediate interactive feedback. Additionally, it enables direct comparison of different headlamp setups and the resulting quality of their beam patterns in different road configurations.

The software uses a smart numerical algorithm to perform its drive simulation, where the vehicle's speed depends on the actual road curvature. With the automatic driving mode, the car always stays on the right (or left) driving path allowing the user to concentrate on the beam pattern, not on the steering of the car. During a simulation, one can switch between the different viewing positions: Driver's View, 20m View (from behind) and Bird's Eye View, which can be selected dynamically during the simulation drive. But not only the car's speed during the simulation is a function of road curvature, but also the driver's view direction and view inclination. The inclined view is typical for simulation of a motorcycle ride. One can also enable or disable the automatic view rotation and inclination. Any general viewing position can be set to accommodate any desired point of view and the simulation can be adapted to left or right hand traffic.

1.3. Road Scenes

LucidDrive comes with a large selection of road scenes, and the ability to create new courses with the "Road Editor" tool by simply editing the road as a sequence of points. From such a polygonal sequence and additional road information (e.g., "four lane highway" or "country side road") to be selected from a broad variety of road types, the software will create a continuous curved road. The created curves adhere to the mathematical form of a clothoid curve. During the drive simulation, all video frames can be automatically exported as bitmaps and in a subsequent process merged into a movie. The user can control the rate and size of the video frame.

1.4. Beam Pattern Analysis

As mentioned previously, the main function of LucidDrive is the beam pattern analysis. The user can switch between various sets of headlamps to verify the differences in resulting beam patterns. For example, the headlamp sets can be a combination of left- and right-aimed low beam lamps. Any number of headlamps with different aiming positions can be combined and mounted on the car. Distance marker lines on the road are included to provide visibility targets for benchmarking.

1.5. Advanced Front Lighting Systems (AFS)

LucidDrive supports dynamic beam patterns such as those in AFS (Adaptive Front-Lighting Systems). The software will dynamically select and adapt the beam pattern according to the current road curvature or radius in a variable look-ahead time window.

1.6. Texture Based Graphics

LucidDrive comes with high, photo realistic and texture-based graphics. The realistic physical model is represented graphically which means that the road-scene luminance, in cd/m^2, is converted into graphics via a complex mathematical model. The model takes surface properties, e.g., reflectivity of the street pavement as well as the scenery (trees, roadside markers and traffic signs) into account. The photo realistic appearance is then accomplished by using textures. The software uses a graphic OpenGL lib and makes use of graphic accelerator hardware. The full functionality can be achieved by employing the latest graphic hardware.

1.7. The LucidDrive Coordinates Systems

The main axis system within LucidDrive can be seen as a reference system. The driving direction is set to the x-axis, the left and right movements on the street perpendicular to the driving direction have been performed in y-direction (positive values to the left, negative to the right), the upper and lower position is set along the z-axis (the resulting axis system is the usual right-handed co-ordinate system).

If one is going to alter the lamp position, one changes the coordinates with respect to the origin of this axis system. Again, x points in driving position, the left or right position of the lamp can be changed with different y-values, and the height of the lamps is controlled via the z-values.

Please note the following points:

- While driving, the road is moving, not the car.
- All distances in LucidDrive (e.g. a mounting position or an offset) are given in [m].
- In contrast to the vehicle axis system, the LIDs (the single beams) are implemented in the usual lighting axis system with +z being the main light propagation direction.

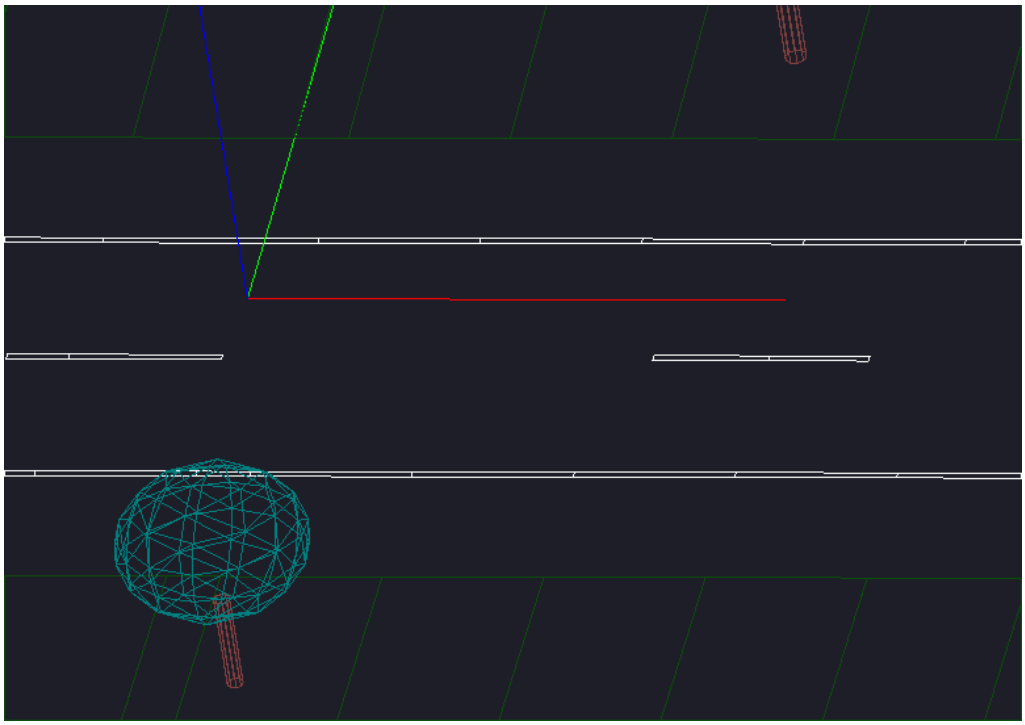


Figure 1.1: This figure shows the axis system of LucidDrive. The driving direction is the x-axis (red), the left or right position on the street is controlled by different y-values (y-axis in green), and the z-axis (blue) points in the upper direction.

Chapter 2. Getting Started with LucidDrive

2.1. The Setup of LucidDrive

2.1.1. Installation

After having received the installation file 'SetupLucidDrive.exe' of LucidDrive via download, you are ready to install LucidDrive. Execute this file and follow the instructions displayed by the install shield program. Next, click onto 'I Agree' to accept the terms of agreement.

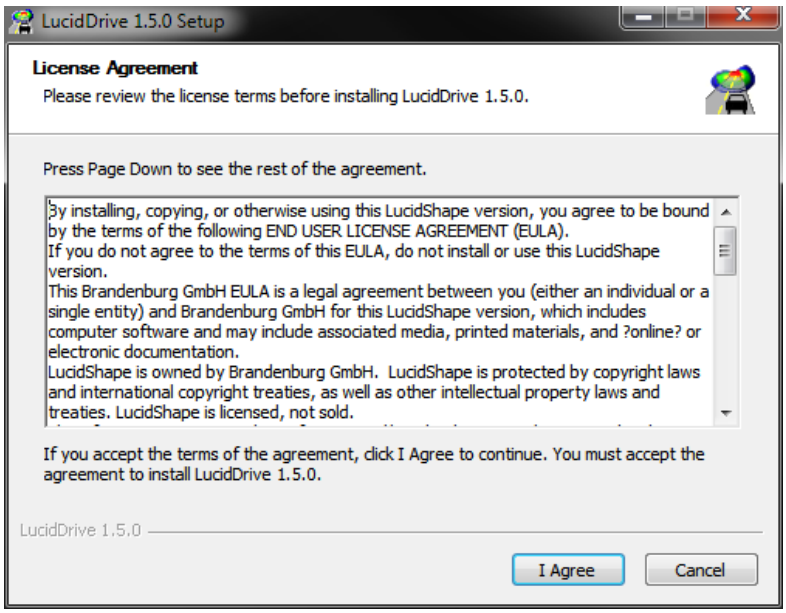

LucidDrive 1.5.0 Setup
License Agreement
Please review the license terms before installing LucidDrive 1.5.0.
Press Page Down to see the rest of the agreement.
By installing, copying, or otherwise using this LucidShape version, you agree to be bound by the terms of the following END USER LICENSE AGREEMENT (EULA).
If you do not agree to the terms of this EULA, do not install or use this LucidShape version.
This Brandenburg GmbH EULA is a legal agreement between you (either an individual or a single entity) and Brandenburg GmbH for this LucidShape version, which includes computer software and may include associated media, printed materials, and ?online? or electronic documentation.
LucidShape is owned by Brandenburg GmbH. LucidShape is protected by copyright laws and international copyright treaties, as well as other intellectual property laws and treaties. LucidShape is licensed, not sold.
If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install LucidDrive 1.5.0.
LucidDrive 1.5.0
I Agree
Cancel

Figure 2.1: The License Agreement Dialog

It is highly recommended to install a new version of LucidDrive into a new folder (i.e. to not overwrite an already existing installation) and to not install it into one of the 'program files' folders. Spaces, dots and digits can be used, so a possible path is 'C:\LucidDrive\v1.5.0\', this also indicates the version in this installation path.

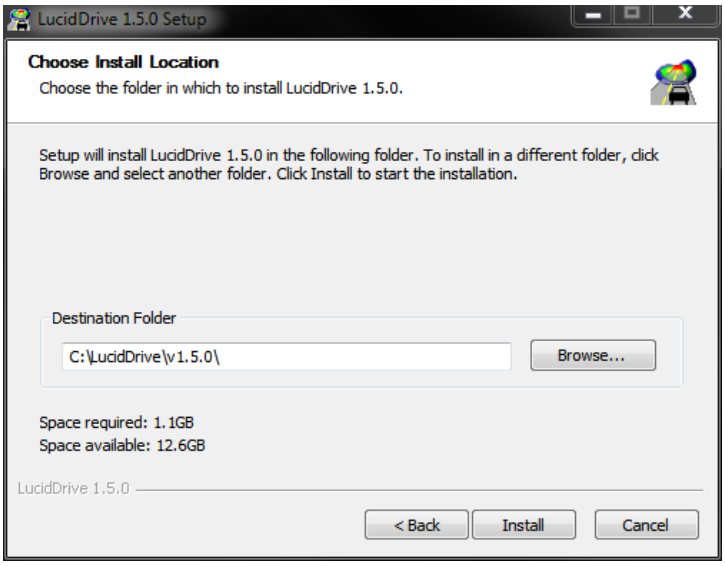

LucidDrive 1.5.0 Setup
Choose Install Location
Choose the folder in which to install LucidDrive 1.5.0.
Setup will install LucidDrive 1.5.0 in the following folder. To install in a different folder, click Browse and select another folder. Click Install to start the installation.
Destination Folder
C:\LucidDrive\v1.5.0\
Browse...
Space required: 1.1GB
Space available: 12.6GB
LucidDrive 1.5.0
< Back
Install
Cancel

Figure 2.2: The installation location Dialog

Click 'install' to install LucidDrive in the chosen directory. After the installation is complete, press 'Finish' to close the installation wizard.

2.1.2. License File Handling

In case you have used your username and password to receive the setup file, the license file automatically is retrieved from our website. If this do not work (e.g. because your computer is located somewhere behind a firewall or similar), please send the license file 'LucidDrive.license' located at '%USERPROFILE%\.Lucid\LucidDrive\' to <info@lucidshape.info>. After having received the activated license, copy it back onto your HDD overwriting the existing one.

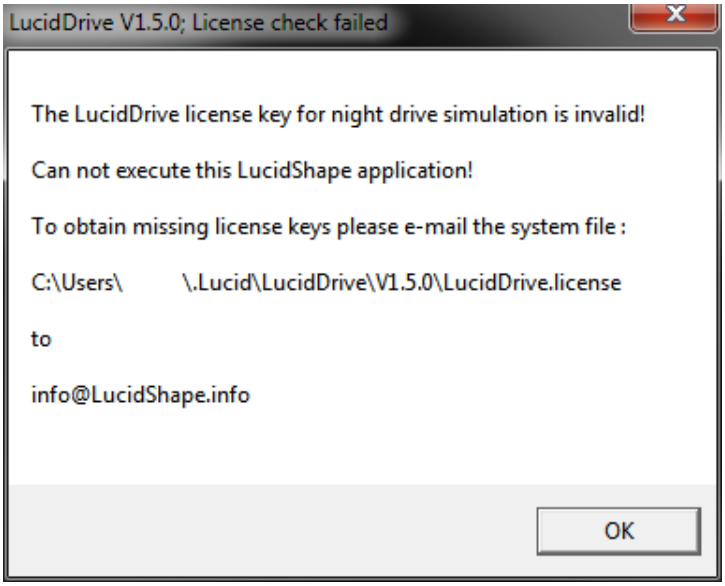

LucidDrive V1.5.0; License check failed
The LucidDrive license key for night drive simulation is invalid!
Can not execute this LucidShape application!
To obtain missing license keys please e-mail the system file :
C:\Users\ \.Lucid\LucidDrive\V1.5.0\LucidDrive.license
to
info@LucidShape.info
OK

Figure 2.3: The license check failed Dialog

After having replaced the license file, start LucidDrive again. You should see an initial screen similar to that in Figure 2.4, “The initial LucidDrive screen”.

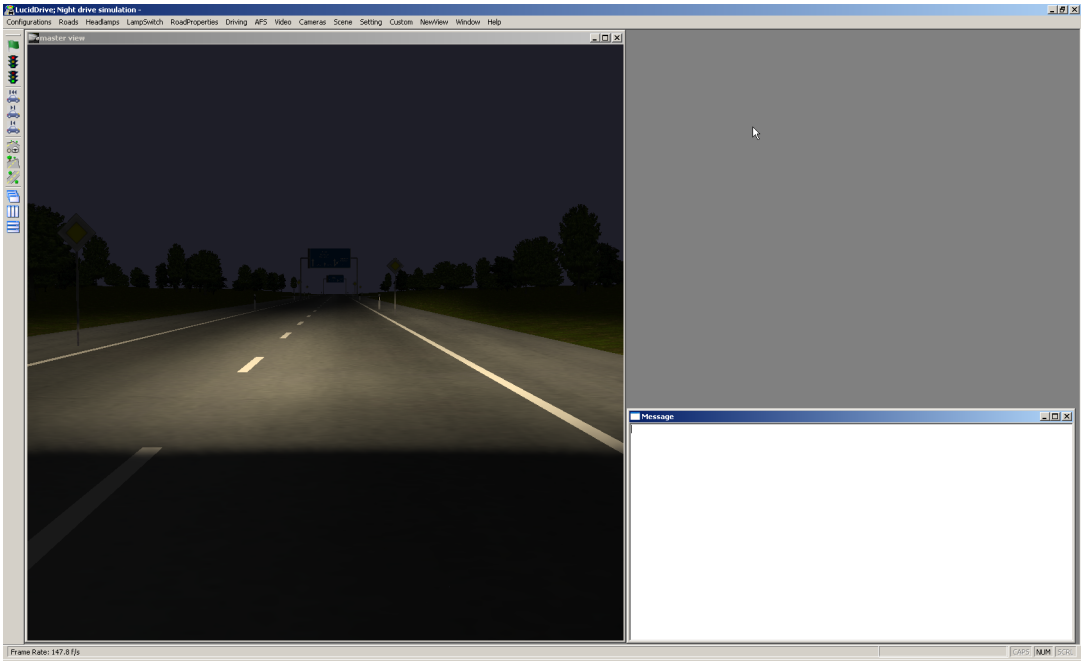


Figure 2.4: The initial LucidDrive screen

2.2. Tool Bar Symbols

In the left upper corner, you can find the tool bar symbols for important shortcuts within LucidDrive. These are:

	Starts a night driving simulation from the beginning. Similar to "Drive" in the driving control dialog box.
	Stops the driving simulation. Similar to "Halt" in the driving control dialog box.
	Continues the night driving simulation from the recent position. Similar to "Continue" in the driving control dialog box.
	Resets the driving scene to the initial start position. Similar to "Reset" in the driving control dialog box.
	Proceeds one time step, i.e. skips the car's position one step in forward direction. Similar to "Next" in the driving control dialog box.
	Skips back one time step, i.e. skips the car's position one step in backward direction. Similar to "Back" in the driving control dialog box.
	Driver view, sets the scene to the position of the driver.
	20 m view, sets the scene to the 20 m view.
	Bird's eye view, sets the scene to bird's eye view, i.e. from top down onto the road.

	Cascades all open views within the LucidDrive window.
	Tiles all open views within the LucidDrive window vertically.
	Tiles all open views within the LucidDrive window horizontally.

LucidDrive tool bar symbols

2.3. Geo View and Message View

The night drive simulation is performed in the so-called Geo View (geometry view). If you close or minimize this window by using the standard Windows commands you can see the message box, along with the menu items and the tool bar.

The message view displays various system messages like loaded textures, files, or error messages and warnings.

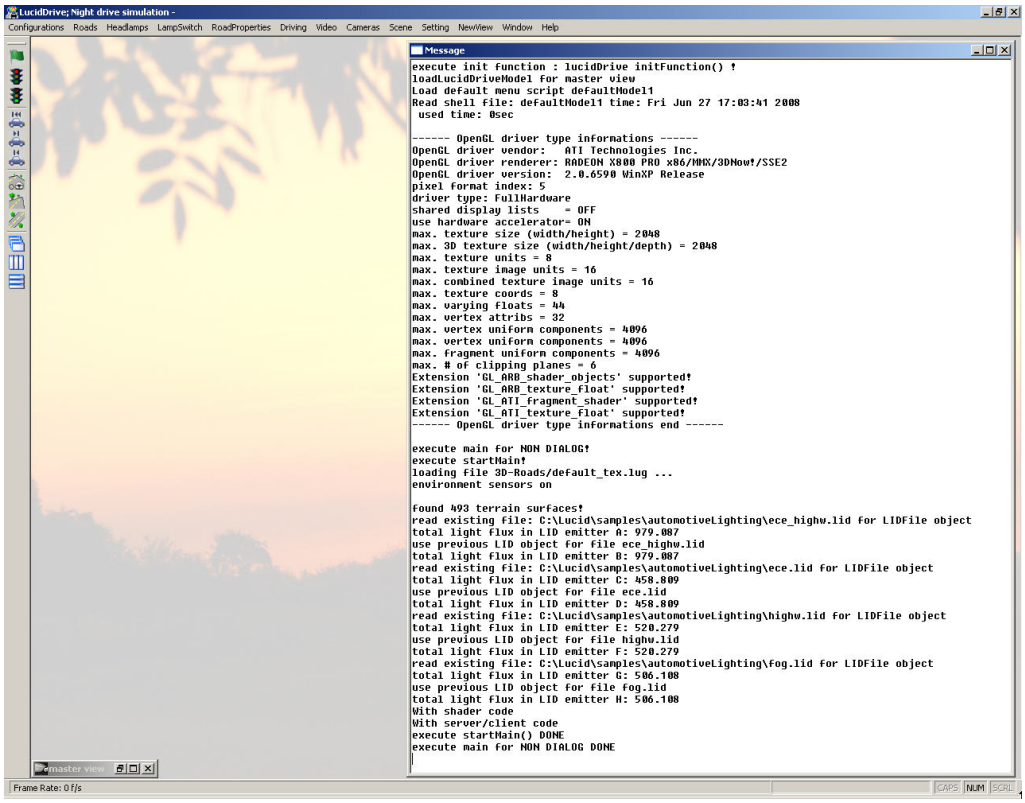

LucidDrive; Night drive simulation -
Configurations Roads Headlamps LampSwitch RoadProperties Driving Video Cameras Scene Setting NewView Window Help
Message
execute init function : lucidDrive initFunction() !
loadLucidDriveModel for master view
Load default menu script defaultModel1
Read shell file: defaultModel1 time: Fri Jun 27 17:03:41 2008
used time: 0sec
------ OpenGL driver type informations ------
OpenGL driver vendor: ATI Technologies Inc.
OpenGL driver renderer: RADEON X800 PRO x86/MMX/3DNow!/SSE2
OpenGL driver version: 2.0.6590 WinXP Release
pixel format index: 5
driver type: FullHardware
shared display lists = OFF
use hardware accelerator= ON
max. texture size (width/height) = 2048
max. 3D texture size (width/height/depth) = 2048
max. texture units = 8
max. texture image units = 16
max. combined texture image units = 16
max. texture coords = 8
max. varying floats = 44
max. vertex attribs = 32
max. vertex uniform components = 4096
max. vertex uniform components = 4096
max. fragment uniform components = 4096
max. # of clipping planes = 6
Extension 'GL_ARB_shader_objects' supported!
Extension 'GL_ARB_texture_float' supported!
Extension 'GL_ATI_fragment_shader' supported!
Extension 'GL_ATI_texture_float' supported!
------ OpenGL driver type informations end ------
execute main for NON DIALOG!
execute startMain!
loading file 3D-Roads/default_tex.lug ...
environment sensors on
found 493 terrain surfaces!
read existing file: C:\Lucid\samples\automotiveLighting\ece_highw.lid for LIDFile object
total light flux in LID emitter A: 979.087
use previous LID object for file ece_highw.lid
total light flux in LID emitter B: 979.087
read existing file: C:\Lucid\samples\automotiveLighting\ece.lid for LIDFile object
total light flux in LID emitter C: 458.809
use previous LID object for file ece.lid
total light flux in LID emitter D: 458.809
read existing file: C:\Lucid\samples\automotiveLighting\highw.lid for LIDFile object
total light flux in LID emitter E: 520.279
use previous LID object for file highw.lid
total light flux in LID emitter F: 520.279
read existing file: C:\Lucid\samples\automotiveLighting\Fog.lid for LIDFile object
total light flux in LID emitter G: 506.108
use previous LID object for file fog.lid
total light flux in LID emitter H: 506.108
With shader code
With server/client code
execute startMain() DONE
execute main for NON DIALOG DONE
master view
Frame Rate: 0 f/s
CAPS NUM SCRL

Figure 2.5: The message view

Chapter 3. The Menu Items in LucidDrive

This chapter describes all commands in LucidDrive by their order of appearance in the pull down menus.

3.1. Configurations

Configurations or config files are files in the native LucidShape format (.lug files) with a set of parameters. Within these files, different configurations or layouts are stored, e.g. different viewing positions, different color mapping or different roads. One can load and save configuration files with LucidDrive. Therefore, it is easy to create a new track with different settings and save it as a configuration file.

Apart from loading and saving configuration files, the pull down menu *Configurations* offers various predefined configurations (see Figure 3.1, "The Configuration pull down menu") which are described in the following sections of this chapter.

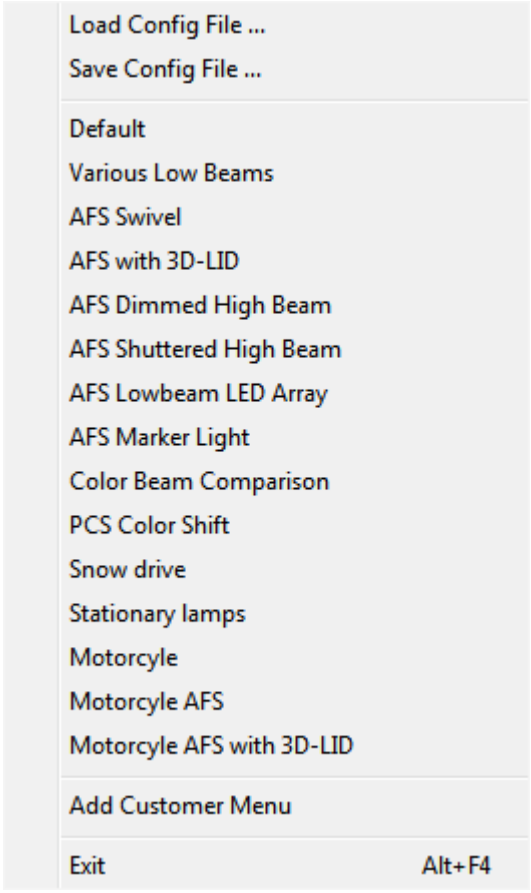


Figure 3.1: The Configuration pull down menu

3.1.1. Load Config File

Loads a configuration file into LucidDrive. The different files contain different parameter. Following configurations can be loaded: 20 m view in false color, AFS (advanced front lighting system), cockpit driver view, bird's eye view (from above), default, driver view, driver view cockpit, driver view cockpit road with high resolution, driver view cockpit sky, and motorcycle view. In the sky or skyline view, a scenery is added besides the road, the motorcycle view enables an inclined view. The inclination is controlled by the road curvature.

3.1.2. Save Config File

After you have changed the setup to meet your individual needs with different parameter, you can save the parameter of this new setup in a file. To load this file, click "Load Config File ...".

3.1.3. Predefined Config Files

Predefined configurations for various viewing situations are available.

Default	The default scene when Lucid Drive is started. It is a simple road with one car and a low beam headlamp.
Various Low Beams	A predefined configuration with different "lousy" low beam pattern for left and right hand traffic.
AFS Swivel	A predefined configuration file which is typical in an adaptive front lighting system. The software will dynamically select and adapt the beam pattern according to the current road curvature in a variable look-ahead time window.
AFS with 3D-LID	The difference to a usual LID is a 3D-LID is used. It shows a 15 degree finger dynamically following the lane curvature. The technique is used here for simulating a car with two headlamps.
AFS Dimmed High Beam	In this example the high beam is connected to a sensor. When an opposing vehicle is approaching, the intensity of the high beam is dimmed down to avoid dazzling the oncoming traffic.
AFS Shuttered High Beam	This example makes use of the plug-in *afsDemoHigh.do*: When approaching oncoming traffic, the two high beam lamps are swiveled to create a gap between the beam patterns on the road.
AFS Low Beam LED Array	A predefined configuration with a low beam LID. It is composed of several LID modules, each representing a LED. It also shows a dynamic 15 degree finger following the lane curvature.
AFS Marker Light	This example makes use of the plug-in *afsDynamicSpot.do*: When reaching a trigger point, a spot beam is illuminating a person or an animal just entering the road.
Color Beam Comparison	A predefined configuration file where different beam patterns, e.g. low beam, fog and high beam are shown with different colors.
PCS Color Shift	This configuration is not an example for AFS but a colored light distribution. It makes use of a spectral calculated LID, furthermore the shutter inside the PCS module is slightly moved for- and backwards to alter (move) the color fringe.

Snow Drive	A predefined configuration file with a snow covered landscape seen from a driver's view position.
Stationary Lamps	A predefined configuration which shows light distributions on the street as caused by stationary lamps.
Motorcycle	A predefined configuration file in motorcycle view. The motorcycle view supports an inclined view, which is typical for a motorcycle ride.
Motorcycle AFS	A predefined configuration file in motorcycle view as above. The difference is that the software will dynamically select and adapt the beam pattern according to the current road curvature in a variable look-ahead time window.
Motorcycle AFS with 3D-LID	A predefined configuration file in motorcycle view as above. The difference is that a 3D-LID is used. It shows a 15 degree finger dynamically following the lane curvature.
Add Customer Menu	This enables one to load a custom made menu, e.g. with only the most used entries or with new options.
Exit Alt+F4	Quits Lucid Drive.

Figure 3.2: Driver's view

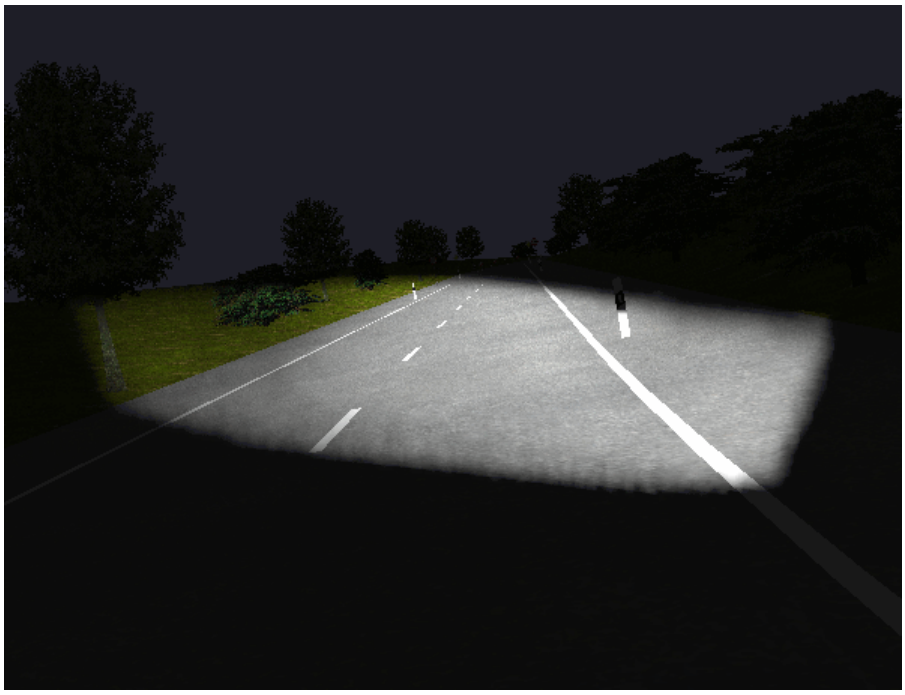


Figure 3.3: Motorcycle view

3.2. Roads

The first entry *Load Roads* offers the possibility to load a road from a file selection dialog box. In contrast to the configuration files, the setup remains the same (besides the different road). Hint: stop the night drive simulation in case you load a new huge road if you don't use the latest PC. Furthermore, various road types both for left and right hand traffic can directly be selected from the drop down menu. The following roads are included in *LucidDrive* and can be found in the roads-folder in the installation path (..\LucidDrive\share\roads\):

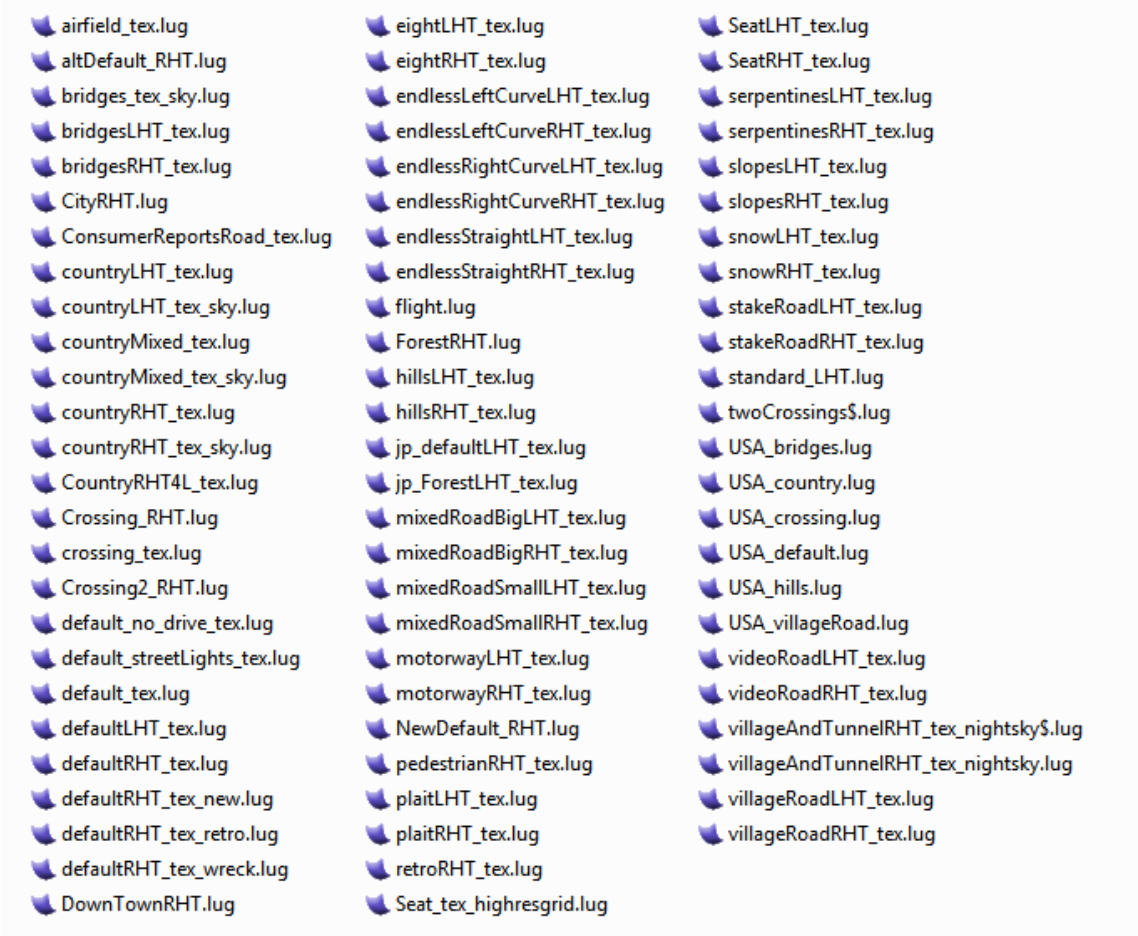


Figure 3.4: File selection box for roads

3.3. Headlamps

This section contains the description of the used options and parameters in the headlamps dialog. In the first subsection the used axis system is explained, the other three handle the three dialog parts in detail. Here one can choose different lamp types, combine two or more lamps to a set (for a direct access in the running simulation), specify the position and the aiming of a lamp or a set.

3.3.1. Main Dialog

This Headlamp Dialog does the complete lamp setup in LucidDrive. Each line in this dialog represents a single headlamp-function, each block defines a group which can be used in the lamp switch dialog. In the picture just below, one block is enfolded by a red line, this block contains three single lamps.

In the following, a 'single lamp' always stands for a line in this dialog, containing either one lamp or a pair of lamps.

The amount and names of the existing vehicles are adopted automatically and are shown in the upper left window of the dialog.

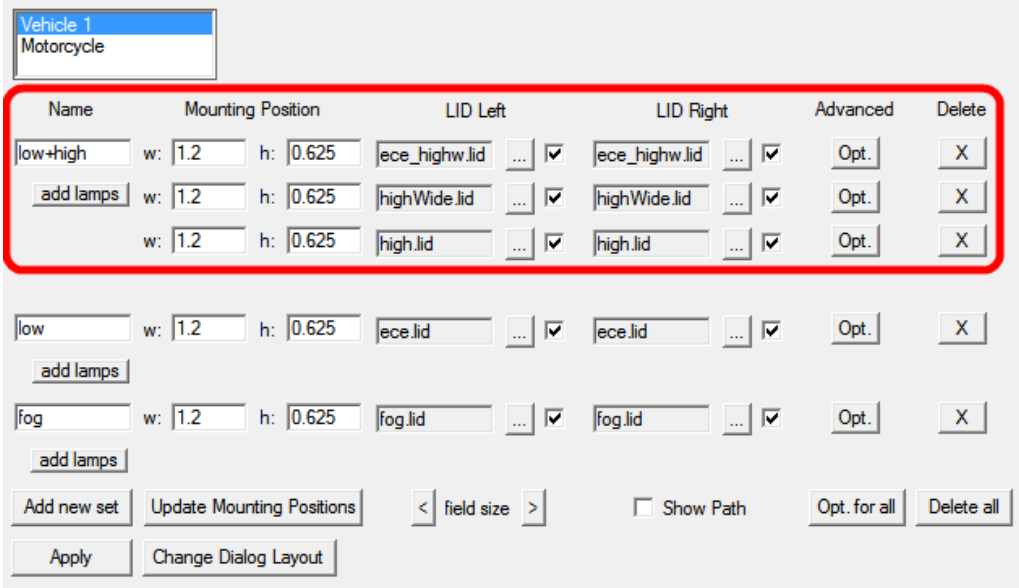


Figure 3.5: The Headlamps Main Dialog

name	The name of the lamp set, this string appears in the lamp switch dialog.
button: add lamps	This will add a line to this set, e.g. to create a combination of lamp functions ('low+high').
mounting position	The width ('w') determines the distance between the left and the right beam. If creating a lamp e.g. for a motorcycle and thus using only one beam, the width has to be set to zero and one of the LIDs (either left or right) has to be deactivated. The height ('h') determines the mounting height of the respective lamp.
LID left/right	Pressing the '...' button opens a file selection dialog, here the LID to be assigned to this lamp has to be chosen. When not using both left and right LID, you can disable one by unchecking one of the two check-boxes.
button: advanced	Pressing this button opens a dialog with additional options and parameters for this single lamp, these will be explained further down this subsection.
button: delete	Pressing this button will delete this line i.e. this single lamp or pair of lamps. A set is deleted when deleting the last line of it.
button: add new set	This will add a new block i.e. lamp set, including a set name, one or more lamps, mounting positions and LIDs.
button: update mountings positions	Updates the changes made for the mounting positions to the currently selected vehicle.
field size	The two arrows allows to enlarge or narrow the input fields.
flag: show path	If unchecked the paths of he LID files are hidden.
button: options for all	This opens the same dialog as for the single advanced options, but the options and parameters will be applied onto all lamps for the currently selected vehicle.

button: delete all	This will delete all lamps sets assigned to the currently selected vehicle.
button: apply	Pressing this button will apply all changes done in this dialog and create the lamps.
button: change dialog layout	If pressed a small additional dialogs opens. Deactivating one of the three flags will include the respective parameters into the main dialog.

3.3.2. Headlamp Advanced Options and Parameters

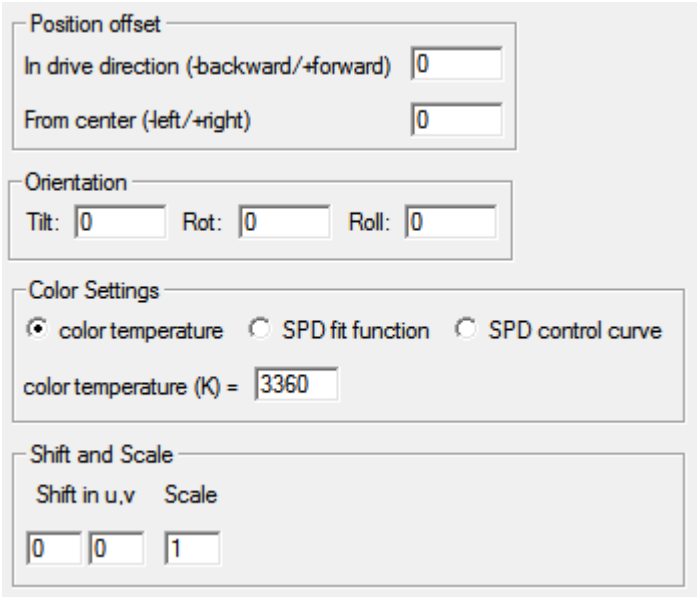


Figure 3.6: The Advanced Options and Parameters of the Headlamps

position offset	Per default the lamps are mounted at x=o and symmetrical to y=o (more precise at y-width and y+width), here an offset to x and/or y can be specified.
tilt	The tilt angle (in degree) of the lamps in vertical direction, a positive value points upwards.
rotation	The rotation angle (in degree) of the lamps in horizontal direction, a positive value points to the left.
roll	The roll angle (in degree), e.g. the inclination of the lamps due to the inclination of a motor cycle in curves.
color	Allows to specify the color of the lamp. Available are color temperature in [K], a manual SPD step function (in a given interval from λ_{min} to λ_{max}) or a SPD control curve which is defined in a file.
shift in u	Determines an optional inclination of the lamps in horizontal direction, a value ›o causes a shift to the right and vice versa.
shift in v	Determines an optional inclination of the lamps in vertical direction, a value ›o causes a shift upwards and vice versa.

scale	Attaches a scale factor onto this lamp, e.g. to take losses into account.

3.4. Lamp Switch

In the lamps selection dialog the specification of the sets done in the dialog *Headlamps* are now available to quickly turn this sets on/off. This can be done either exclusively or more than one simultaneously. Switching the flag *select exclusive* toggles the selection between radio buttons and flag buttons.

Figure 3.7: Lamp Selection Dialog

The use and setup of headlamp sets when using more than one vehicle is described in detail in the next chapter.

3.5. Visual Properties

In this dialog among other, the intensity mapping and coloring properties of the road can be modified. In addition some graphical help like isolux lines can be activated.

luminance range	Sets the min and max values for the luminance range in [cd/m^2].
button: update min/max	Applies the current settings to the geometry view.
flag: fix range	If unchecked the button *update min/max* replaces the current values with the default ones. This can be used e.g. to compare lamp sets with different light fluxes.
background color	Sets the color of the background i.e. the sky.
ambient light color	Sets the color of the ambient light, this can be set independent to the background color.
luminance [cd/m^2]	Determines the amount of the ambient light.
brightening factor (old)	This parameter was used in older versions of LucidDrive to brighten up everything on the screen.
radio button: diffuse BRDF type	Sets the type of the diffuse BRDF (Bidirectional Reflectance Distribution Function) to either Lambertian, Hoffmann, Nakata or Lommel-Seeliger. *Lambertian* is the basic type, the other three use special empiric models for the calculation. (In the file ../luciddrive/shader/custom.frag the more skilled users can modify the used parameters and equations.)
radio button: specular BRDF type	Sets the specular part of the BRDF behaviour to either none or Cook-Torrance (alternative name Torrance-Sparrow). (This is independent from the diffuse BRDF setting.)
diffuse ratio	Determines the ratio of diffuse to specular BRDF. If e.g. a value of 0.15 is assigned onto the road surface, this strongly reduces

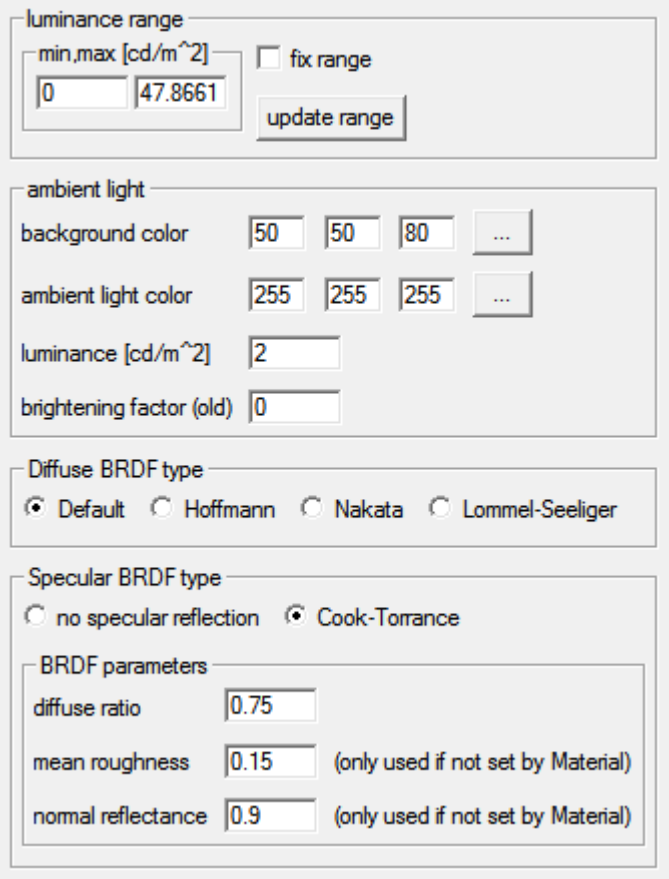


Figure 3.8: Visual Properties Dialog (I)

	the amount of scattered light relative to the ideal reflected light thus the road appears rain-wet.
mean roughness	Specifies the roughness of the surface, thus the reflection of the light and thus the amount of light being scattered back to the driver. (A perfect specular plane would reflect no light back to the driver, the road would appear almost completely dark.)
normal reflectance	When perpendicularly illuminating a surface, this values gives the amount of light which is reflected back to the source without any scattering.

For a more detailed description regarding the different BRDF types, please refer also to Section 8.2, “BRDF”.

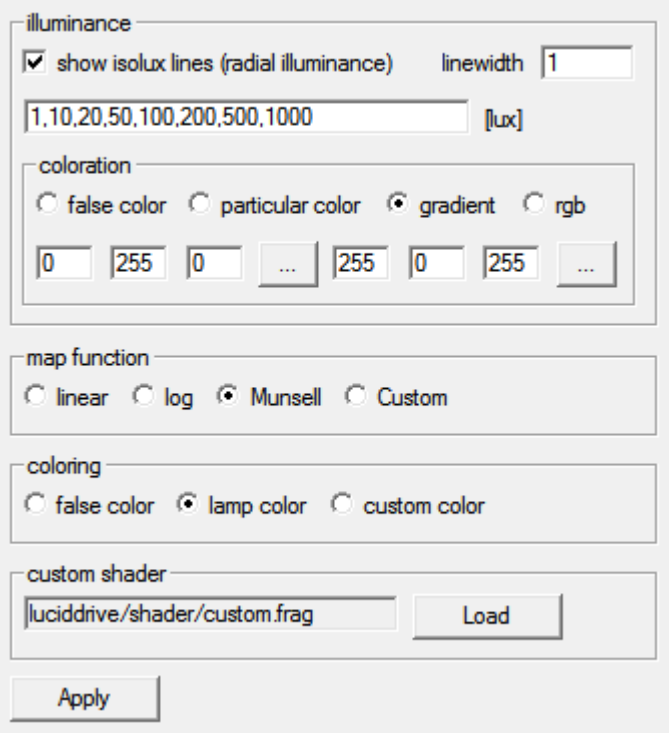


Figure 3.9: Visual Properties Dialog (II)

flag: show isolux lines	If checked the below entered isolines are shown in the different views after pressing *Apply*. (Please note: *radial* means that the normal on the measuring device always points directly to the driver, in difference to *horizontal* and *vertical*.)
input window	Here one may enter a set of values for the wanted isolines in [lux].
line width	Determines the width of the isolines.
colorations	The color of the isolines can be chosen among four types. • false color: The color coding of the isolines goes from red (highest intensity) over yellow and green to blue (lowest intensity). • particular color: All isolines are drawn in the specified color. • gradient: Analogue to the case of *false color*, but here one can define both the starting and the ending color (again in the order highest → lowest intensity). • RGB: The isoline with the lowest intensity is drawn in red, the second lowest in green, all other in blue.
radio button: map function	Sets the intensity mapping function to either linear, logarithmic, Munsell or custom (the appearance of *Munsell* is something between linear and logarithmic, this is almost the way it is sensed by the human eyes). The latter allows the user to use a self-defined display mode. For this purpose, the custom-files in the folder /samples/automotiveLighting/shader have to be modified.

radio button: coloring	Sets the road coloring properties to either false, lamp or custom color.
button: Apply	Performs the changes of the intensity mapping and coloring properties.

Concerning the intensity mapping function, please refer to the LucidShape Technical Reference for a more detailed discussion.

The two parameters *mean roughness* and *normal fresnel* do not overwrite the already assigned values to a surface (if e.g. already done in *RoadEditor*).

3.6. Driving

The driving control dialog box sets the driving properties, starts and stops the car, resets the driving scene and controls the velocity. It also sets the rotation of the driver view and enables the possibility to record the simulation. The software uses a smart numerical algorithm to perform its drive simulation, where the vehicle's speed depends on the actual road curvature. With different "look ahead" values, the actual speed can be altered, i.e. the velocity will be reduced before the curve, and the acceleration will start before the curve ends. But not only the speed depends on the curvature of the road, also the optional automatic rotate view, the automatic roll angle, the roll lamps angle and the dynamic beam patterns (AFS).

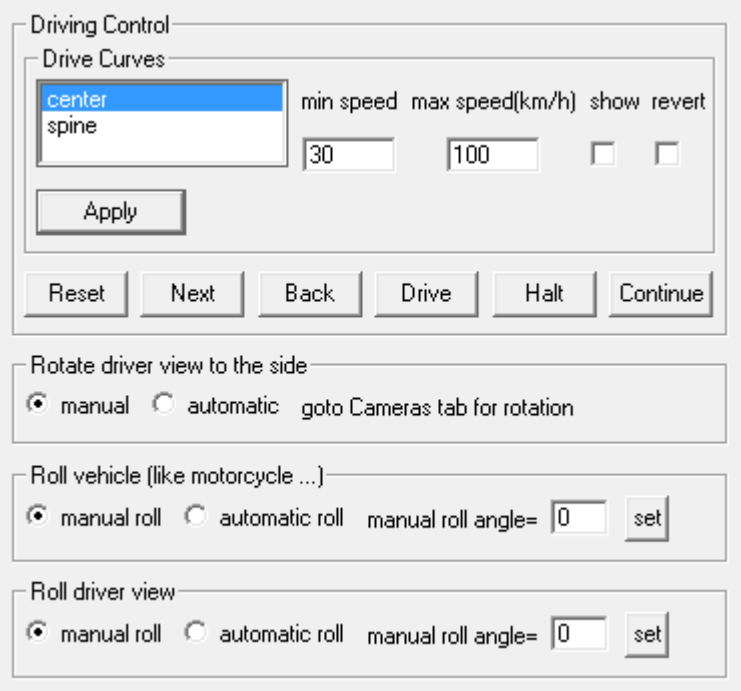


Figure 3.10: The Driving Dialog

drive curve	This switch sets the driving curve to either center of the road or spine (driving lane).
min/max speed	Specifies the minimum and maximum speed in km/h.

flag: show	Shows/hides a specific driving lane on the road. For each one has its own flag, more than one line can be shown, e.g. to compare the driving curves.
flag: revert	If checked the driving curve is reversed (the car is driving 'backwards').
button: apply	Applies the changes made on both the velocity and the two flags.
button: reset	Resets the driving scene to the initial start position.
button: next	Applies the next time step.
button: back	Applies the previous time step, i.e. the simulation makes one step back.
button: drive	Starts the driving simulation from the initial starting pint.
button: halt	Interrupts (but not ends) the driving simulation.
button: continue	Continues the simulation from the last time stamp.
radio button: roll vehicle	Sets the rolling behaviour to either manual (with a fixed angle) or automatic. If set to "automatic roll", the lamps are vertically inclined which is typical for a motorcycle ride, the roll angle depends on the road curvature. If set to manual, a fix lamp roll angle is applied.
radio button: roll driver view	If set to "automatic", the driver view is vertically inclined automatically. The inclined view is typical for a simulation of a motorcycle ride. The actual inclination depends on the road curvature. If set to manual, a fixed driver roll angle is applied.

3.7. Scripts/Plug-ins (Vehicle Dynamics, AFS, Aux Plug-ins, Trigger Points)

LucidDrive provides several predefined scripts, sorted into four different groups. These plug-ins can be used to modify the driving behaviour, the handling of some or all LIDs (orientation, behaviour like e.g. dimming), activate additional plug-ins or to trigger certain events while driving along the track.

Each of the next four menu items will open the corresponding dialog box so select among the scripts. The scripts can be activated by just clicking on the respective entry in the list and deactivated by pressing *remove*. Pressing *show* will open a small dialog, here some options and parameters can be altered (depending on the respective plug-in).

For further information about the concept of plug-ins and trigger points in LucidDrive (including their syntax, data exchange, the respective dialogs, etc.), please also refer to Chapter 5, *Scripts, AFS and Vehicle Dynamics Plug-ins* or Chapter 7, *Plug-ins provided in LucidDrive*.

3.8. Video

Allows to make a video from the driving simulation. Specify an output directory for the single pictures to be saved at, a frame rate and select the output file format. These pictures later can be used to make e.g. an avi movie.

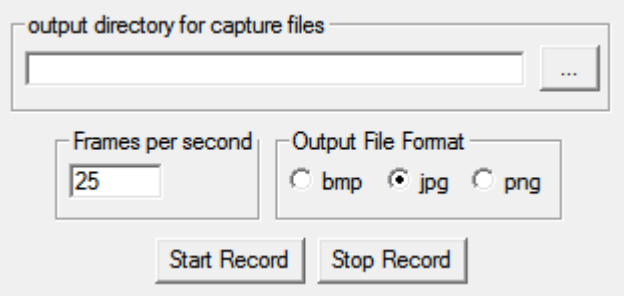


Figure 3.11: The 'Video' Dialog

output directory for captured files	Exactly that.
frames per second (fps)	The frame rate equals the number of pictures created per second.
start record	Starts the recording of the simulation. The recording is only active during driving. Therefore *Start Record* and *Drive* have to be pressed both to start the record.
stop record	Stops the recording of the driving simulation.

3.9. Cameras

Performs the general camera settings of LucidDrive. The different views (driver view, 2om view, bird's eye view and pedestrian view) refer to the center of the same coordinate system. The driving direction is set to the x-axis, the left and side movement on the street perpendicular to the driving direction is in y-direction, and the upper and lower positioning is achieved along the z-axis. While driving, the road is moving, not the car. If one is going to alter the driver's height for example, one has to change the z-value in the *driver view position* dialog box entry. Target defines the distance on the road the driver is looking at.

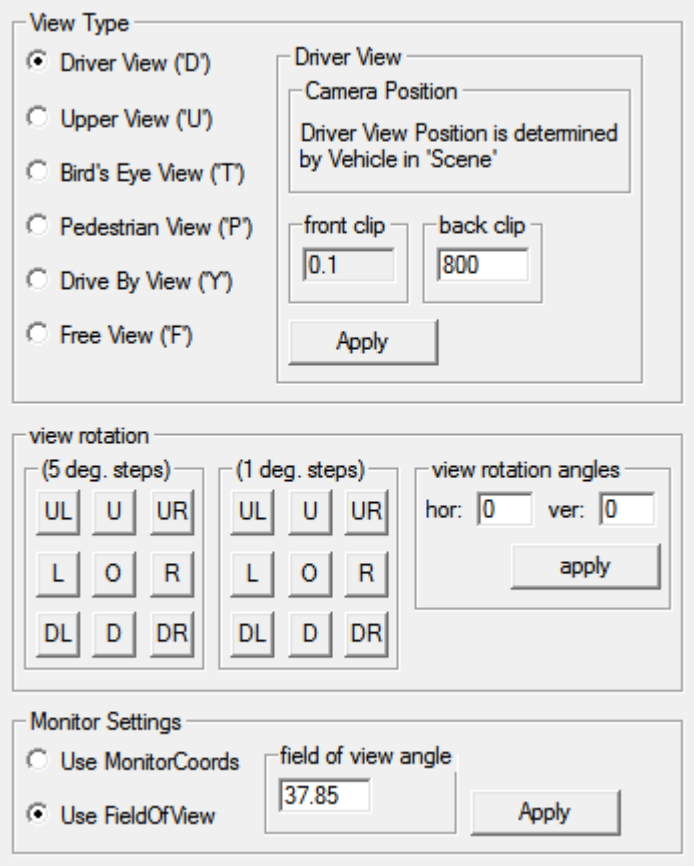


Figure 3.12: The Camera Dialog

view type	One can toggle between six different views either by selecting by a mouse-click oder by pressing the respective letter on the key-board. All changes done to the parameters will be performed when pressing *Apply*. All position values are given in [m].
position and target	Specifies the respective view and the target position, all given in [m].
DriveBy parameters	Sets the drive-by parameter as the number of positions during a loop and the offset, i.e. the distance to the car.
free view	Here the camera position is fixed, specified by the entered position and orientation.
view rotation	The view rotation can be performed by pressing some predefined buttons, steps of 5° or 1° can be chosen.
view rotation angles	The two values show the current angles in horizontal and vertical direction respectively. These angles can either be entered directly or changed via the just above described buttons.
radio button: monitor settings	Switches the monitor settings to either monitor coordinates or field of view.
monitor start/end x/y	Specifies the starting/ending coordinates of the monitor, this resizes the view of the scene.

field of view angle	Specifies the used angle of vision (the smaller the angle, the closer the look onto the road and fewer objects in the outside).

3.10. Scene

Includes scene elements for the driving view, e.g. an aiming wall or a grid on the road.

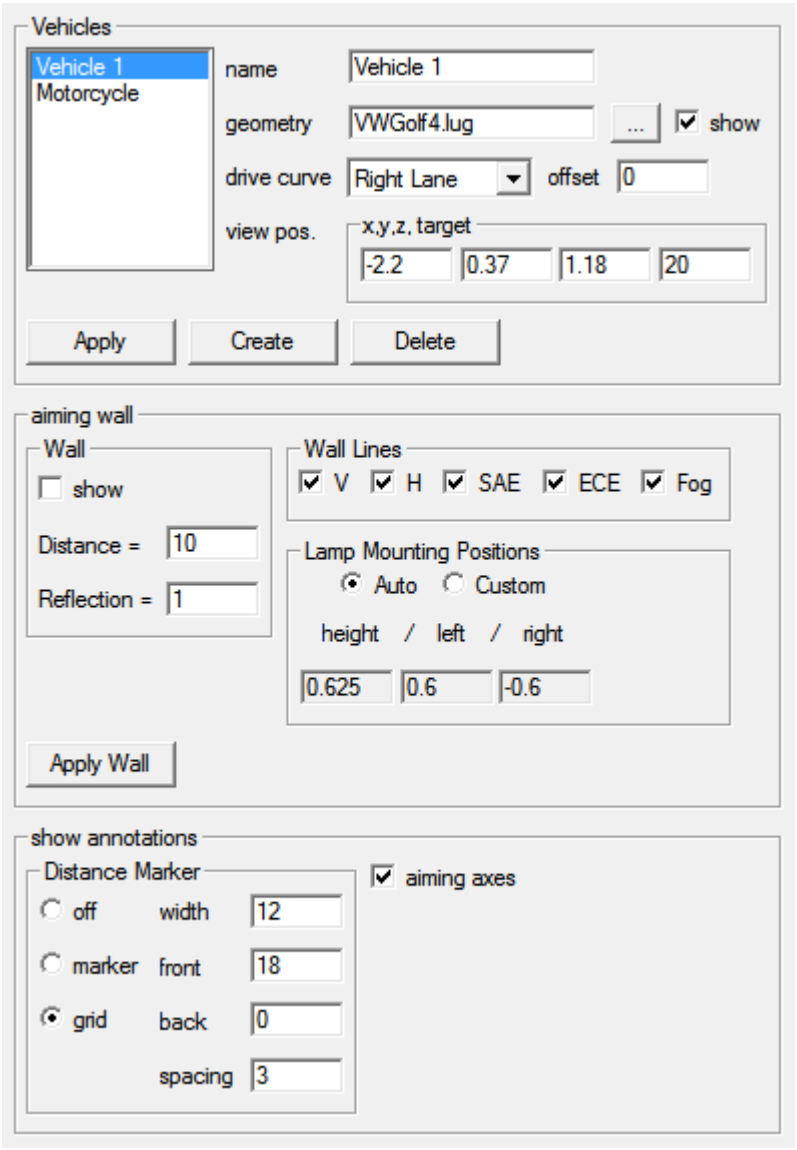


Figure 3.13: The Scene Dialog

vehicles	This dialog part enables one to select, load, create and modify vehicle geometries. One can give unique names, choose a provided model to be shown in the driving view and specify the driving curve via a pull down menu (e.g. left lane, left passing lane, center, ...). The offset determines the starting position on the road (in [m], e.g. if more than one car is on the track).

viewing position	Sets the coordinates for the driver view. Changing the sign of the y-value will switch between RHD and LHD (right-/left-hand driving), the z-value maybe needs to be verified for different cars (→ seat height). The target is the point on the road the driver is looking at (distance in [m] relative to the driver's position).
flag: aiming wall	An aiming wall may be shown which is defined by a distance (car to wall) and the reflection scale (again a value ›1 is possible).
flag: wall lines	Checking (unchecking) the single flags will show (hide) the respective lines on the wall.
radio button: lamp mounting positions	If set to custom, the mounting position of the lamp can be changed.
radio button: show annotations	Here one can specify whether markers shall be shown or not and the type of. *Off* shows no marker at all, *marker* shows several differently colored distance lines. *Grid* shows the driving lane and some marker lines, specified through the entered parameters. (The grid is calculated as quadratic, in order to show distance lines the width has to be at least twice as spacing.)
flag: aiming axis	If checked an usual right handed axis system is shown.

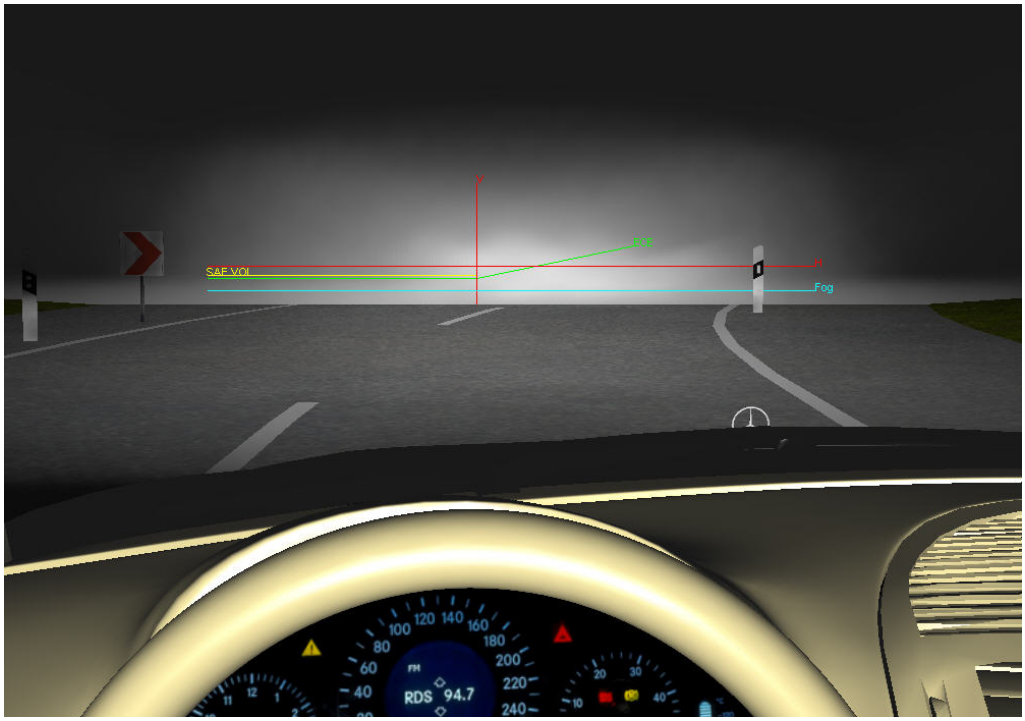


Figure 3.14: Driver view with the aiming wall and the lines for different beam pattern regulations (SAE VOL, ECE and FOG lamp).

3.11. Settings

Performs some general settings

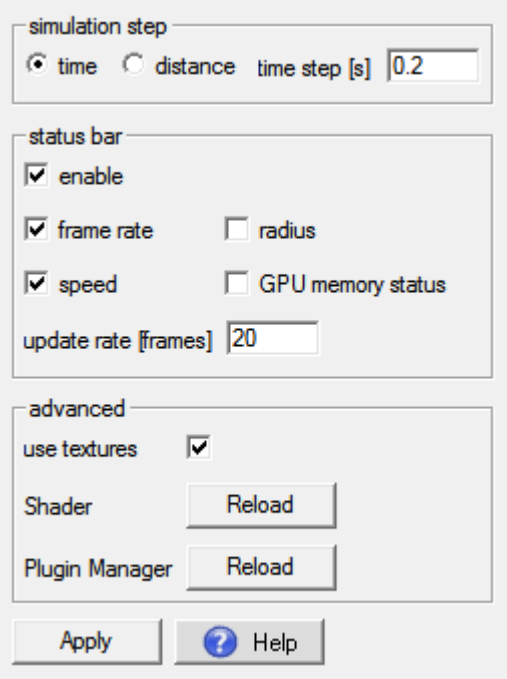


Figure 3.15: The Settings Dialog

radio button: simulation step	Depending on the choice made by the radio button, either a time in [s] or a distance in [m] can be entered to determine the length of a single simulation step. Increasing this value too much will lead to kind of hopping/stuttering of the vehicle along the road.
flag: status bar	If checked, some informations may be shown in the status bar (left lower corner on the screen).
frame rate	If checked, the actual frame rate is shown in the status bar.
speed	If checked, the actual speed is shown in the status bar.
radius	If checked, the actual radius is shown in the status bar.
GPU memory	If checked, the current memory status of the GPU is shown in the status bar.
update rate [frames]	This is the rate (i.e. frames per second) the informations in the status bar are updated.
flag: use texture	Switches the use of all textures on/off.
flag: use shader	Switches the use of a shade on/off.
buttons: reload Shader /Plug-in Manager	If something went wrong i.e. an error occurs, these two buttons give the possibility to completely reload the respective object.

3.12. Miscellaneous Menu Commands

Some windows related command

new view	Creates a new driving view with the default settings.
windows → cascade	Arranges all open windows in a cascade.
windows → tile vertical	Arranges all open windows in vertical tiles.
windows → tile horizontal	Arranges all open windows in horizontal tiles.
windows → arrange icons	Arranges the icons.

Chapter 4. Multiple Cars

Beside the simulation of one car and its headlamps one further main task of LucidDrive is to have multiple vehicles on the track to have some opposing traffic. This is used to evaluate not only the effect of the own lamps onto the geometry but to have an impression of the own headlamps when sitting in another vehicle or being a pedestrian.

In case there are more than one vehicle on the road, at any time only one car is active (i.e. marked in the vehicle list in the scene elements dialog). This active car can be controlled directly, both the headlamps and the driving (either manually or automatically driving).

The remaining vehicles are set as passive, i.e. they are controlled by LucidDrive directly. When switching to another vehicle, the at last used lamp set stays active until we switch back to this vehicle.

4.1. Other Settings

Beside the headlamp set(s) some other parameters of the cars can be specified. The speed is not controllable for each passive car separately but for each lane, this can be done in the usual driver dialog.

Further options can be set in the scene elements dialog, all of the options in the upper part of the dialog are independent from each other (so e.g. one geometry file can be used for more than one vehicle, etc.). The vehicles can be named and a geometry can be assigned to. This geometry can be switched on and off by checking the flag "show". The flag "lit" only controls the visibility of the car itself, it has no influence to the headlamps. Furthermore the driving curve and the view position inside the car can be specified. The parameter "offset" controls the starting position for the respective car on the track in [m], thus effecting the starting distance between the single cars. If choosing different speed limits, these distances of course are not fixed during the simulation.

The three buttons "apply", "create" and "delete" will apply the settings, create a new vehicle or delete the marked one. The options and parameters in the lower part are again identical for all vehicles on the track, the usage is the same as already described in chapter 3 above.

Chapter 5. Scripts, AFS and Vehicle Dynamics Plug-ins

This chapter describes the script architecture in *LucidDrive*, this can be used for implementing custom plug-ins like e.g. AFS behavior and Vehicle Dynamics for semiautomatic and manual driving. It consists of a mechanism to load plug-in scripts written in LucidScript which implement this behavior and an API that the scripts can use to get specific LucidDrive data necessary for their calculations. Furthermore the possibility to call external code through a COM interface is described, so scripts can delegate their work to in-house developed or already existing applications.

5.1. Introduction

In the version 1.2.1 of LucidDrive user scripting of vehicle dynamics and adaptive lighting behaviour was introduced. With this feature, users can implement their own solutions tailored to their needs and products. In order to fully utilize the potential of this script plug-in architecture, one has to understand the interface of the plug-in scripts to LucidDrive and the possibilities to interface from the script to external code.

In the version 1.4.0 the concept of plug-ins in LucidDrive was expanded and redesigned. Now there is a kind of distinguish between 'primary' plug-ins like e.g. AFS and auxiliary ('secondary') plug-ins like e.g. graphical elements (tachometer and rotation speed indicator) or a data logger. Also there are now so called Trigger Points available, when crossing these points on the track during the simulation, predefined actions will be released. Furthermore there is now the possibility of data exchange between different plug-ins to ensure the modularity of the plug-in-system.

First we describe the necessary conventions used in LucidDrive for defining the car and lamp position and orientation. Then we give an overall picture of how script plug-ins are loaded and when they are executed in the LucidDrive pipeline. Next, the interface functions are listed. In the end, the example scripts provided in the LucidDrive installation are listed and shortly described to give the reader an impression of their capabilities.

5.2. LucidDrive Conventions

For describing the vehicle position and orientation LucidDrive uses a vehicle coordinate system (VCS) which follows the conventions described in [1] and which is shown in Figure 5.1, “Car coordinates used in LucidDrive”. The geometry of the car is transformed to it and the position and orientation of the lamps are specified relative to this system. The origin of the VCS does not need to be in the center of gravity of the car (in the LucidDrive examples it lies in the centre between the headlamps).

The lamp position and orientation is specified using a right-handed lamp coordinate system (LCS) which follows the convention in LucidShape. The angular coordinates of the light distribution are mapped to this coordinate system. This system is relative to the VCS.

5.3. Plug-in Classification and Pipeline

The scripts are written in the script language of *LucidShape* respectively *LucidDrive*. Each one can implement a dialog for manipulating parameters of the script.

The simulation of LucidDrive operates in single steps, the simulation intervals of these steps are determined by the given increment, this can be either in time or in space. The calculation sequence of the single parts is always the same:

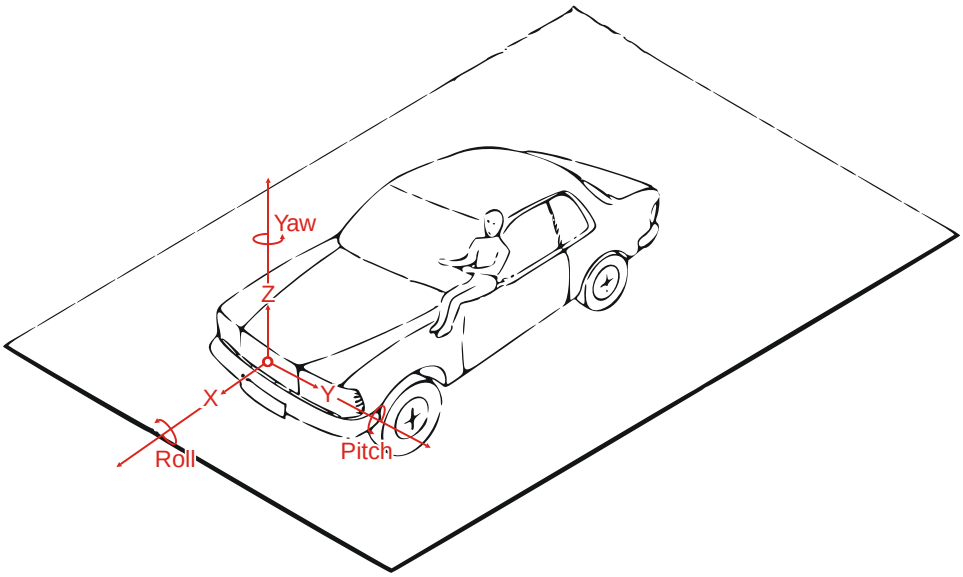


Figure 5.1: Car coordinates used in LucidDrive

(1) Update Inputs → (2) Calculation of Vehicle Dynamics → (3) AFS Plug-ins → (4) Auxiliary Plug-ins.

Next the order of the calculation steps and the steps itself are explained, in the next but one section the architecture of the plug-ins is explained.

1. At first all inputs to the vehicle are taken into account. This can be both deterministic (in the automatic driving mode the driving lane and the speed are pre-given) and user-defined (in the manual driving mode the received inputs come from the steering angle of the wheel and the position of the two pedals).

   LucidDrive does not have a perfect physical model, but it calculates the on the vehicle acting forces quite detailed. The input parameters are the actual gear, the rpm of the engine, the incline of the road, the mass of the car and the position of the two pedals.

   Also implemented is a model that calculates an incline of the car due to the actual positive or negative acceleration via the pedals → pitch (this gives the possibility to test an optional dip or dim control of the lamps).

2. With the actual VCS and the dynamic inputs the next simulation step is calculated (temporal or spatial). It is very important that the chosen interval is small enough to ensure a smooth motion of the car, both for the view of the light and the motion and the VCS of the car itself, please see also Section 3.11, “Settings”. After computing the simulation step, both the VCS and LCS are updated ('the car drives forward'), now the static light is calculated (without any dynamical effects like AFS).

3. The next step in the pipeline is the AFS plug-in. This plug-in receives the wanted data from LucidDrive (among other things there are the actual position on the driving lane and its bending, the yaw angle of the car and the speed) to compute the swivelling angle of the AFS attached beams. The complete light (both statical and dynamical parts) is projected into the scene and can be seen on the monitor.

4. The group 'auxiliary plug-ins' contains any further active plug-in. The possible duties can cover a wide range of functions, e.g. to control trigger points, add an audio output (basic simulation of the engine noise), logging certain parameters, etc.

5.4. Plug-in Architecture

1. -

2. Among other things the Dynamic plug-in contains function calls like

```
updateDynamics();
setVehicleAxis(…);
setSpeed(…);
```

   When the forces have been computed the acceleration, the velocity and distance can be calculated

$$\begin{aligned} a &= \frac{F(\dots)}{m} = \frac{dv}{dt} \\ v_{n+1} &= v_n + \int a\,dt \approx v_n + a \cdot \Delta t \\ s &= \int v\,dt \approx v \cdot \Delta t \end{aligned} \tag{5.1}$$

3. The two default plug-ins which are provided in LucidDrive are *modifyLight()* for the automatic driving mode and *modifyLightManual()* for the manual driving mode /›).

4. One group of possible auxiliary plug-ins are the so called trigger points. When reaching a certain position on the track a predefined action is started. More precise there is a 'trigger circle' around the place the animation will take place, the default radius is 10m (this can be specified in the trigger plug-in). This is to ensure the car hits the trigger even in the manual driving mode, in the automatic driving mode the trigger point is a one-dimensional coordinate on the driving lane.

   The triggered action e.g. can be a person on a cross-walk, a wild animal appearing at the street or in general any person/animal/object crossing the road. To prevent this object from permanently crossing the road (which will look very unrealistic) and safe calculation time, this wanted action is only released via this trigger point.

   Other possible groups are plug-ins to add graphical or acoustical elements to enhance the feeling of the speed during the driving simulation. Furthermore there can be plug-ins to log some or all options/parameters to the HDD every few seconds, this gives the possibility to do further analysis on the data, especially after a non-deterministic manual drive.

5.5. Signal-Slot Mechanism

With the help of two short exemplary program routines (an AFS plug-in and a Logger plug-in), the data exchange in *LucidDrive* will be explained.

The data exchange will not be done by the plug-ins themselves but will be provided by LucidDrive. Each single plug-in has a list of function calls and variables. When starting the simulation LucidDrive is comparing these lists, if there is an identical function/variable in two or more plug-ins, LucidDrive will provide the data exchange according to the request or allocation calls of these data. Due to this concept the correct spelling of these functions/variables is quite important.

To prevent the user from being asked every time two objects in different plug-ins have similar but not identical names, there are no checks about similar spellings and thus no error messages will occur.

AFS Plug-in:

```
double swivelangle;
modifyLight();
...
double swivel_signal(){
   return swivelangle;
   ...
}

object getSignals(){
   return ("swivel");
}

object log_signal(){
   return (swivel, ...)
   ...
}
```

Logger Plug-in:

```
double sa;
update();
...
cout « ...
...
void log_slot(object content);
...
void swivel_slot(double angle){
   sa = angle;
   ...
}

"swivelangle is: ", swivel ...;
```

Most important are the three underlined objects, in this case 'swivel' (in general they can be variables, function calls, etc.), if they are identical, LucidDrive will detect that and will provide the data exchange.

Second example for a logger Plug-in:

```
#include <windows/winresrc.hdo>

object gs_keyObserver = ObserverTrigger("observer for key events", KeyEvent("key clicked"),
"onKeyPressed", ($) );

double onKeyPressed(object event) {
double downUp, key, repeatCount;
event.getKeyData(downUp, key, repeatCount);

cout « "key " « key « " pressed." « endl;

return 0; // 0 == not handled, 1 == handled, do not propagate
}
```

This last code can be integrated directly into the logger.do plug-in (to call 'appendToLog' there) or into any other plug-in.

In a visualized vision each plug-in has to be seen as a box with certain sockets on it, if two boxes have a pair of identical sockets (i.e. the name of an object), LucidDrive provides a connection between those boxes in the specified direction (depending on allocation or request).

Chapter 6. Plug-in Script Interface

6.1. Lamp functions

These functions allow to get the state of the lamps which are loaded in LucidDrive. It is possible to modify the axis system of a lamp or the 3th and 4th parameter of a 4D light distribution. At least one of them has to be modified in order to manipulate the lamps.

The following functions are listed in alphabetical order.

- object getLamp(double idx);

  returns the idx-th lamp as an object of type PointLight
  - return the idx-th lamp
  - param idx index of lamp ranging from 0 to getNumLamps()

- object getLampIndicesOfSet(double idx);

  returns the lamp indices of the idx-th lamp set, so they can be fetched with getLamp
  - return the lamp indices of the lamp set as a list
  - param idx index of lamp set ranging from 0 to getNumLampSets()

- object getLampLID(double idx);

  returns the light distribution of the i-th lamp as an object of type LIDEmitter
  - return the light distribution of the idx-th lamp
  - param idx index of lamp ranging from 0 to getNumLamps()

- object getLampRelativeAxis(double idx);

  returns the axis system of the i-th lamp as an object of type AxisSystem
  - return the axis system of the idx-th lamp
  - param idx index of lamp ranging from 0 to getNumLamps()

- string getLampSetName(double idx);

  returns the name of the idx-th lamp set
  - return the name of the lamp set
  - param idx index of lamp set ranging from 0 to getNumLampSets()

- object getLampsOfSet(double idx);

  returns the lamps of the idx-th lamp set
  - return the lamps of the lamp set as a list
  - param idx index of lamp set ranging from 0 to getNumLampSets()

- double getNumLampLids();

  returns the number of loaded light distributions
  - return number of loaded light distributions

- double getNumLamps();

  returns the number of loaded lamps
  - return number of loaded lamps

- double getNumLampSets();

  returns the number of lamp sets, i.e. collections of lamps to be switched simultaneously
  - return number of lamp sets

- double isLampOn(double idx);

  returns true if the idx-th lamp is activated
  - return true if lamp is activated
  - param idx index of lamp ranging from 0 to getNumLamps()

- double isLampStationary(double idx);

  returns true if the i-th lamp is stationary, i.e. the axis is not relative to the car axis system, but to the global axis system
  - return true if lamp is stationary
  - param idx index of lamp ranging from 0 to getNumLamps()

- double setLampAxis(double idx, object axis);

  sets the axis of the idx-th lamp
  - return true if successful
  - param idx index of lamp ranging from 0 to getNumLamps()
  - param axis the new axis to be set

- double setLampDimming(double idx, double dimFactor);

  sets the dimming of the idx-th lamp to dimFactor, i.e the intensity of the lamp is multiplied with this factor
  - return true if successful
  - param idx index of lamp ranging from 0 to getNumLamps()
  - param dimFactor the factor of dimming between 0 and 1

- double setLampLIDDataBlock(double idx, double iw, double il);

  sets the data block of the xD LID of the idx-th lamp
  - return true if successful
  - param idx index of lamp ranging from 0 to getNumLamps()
  - param iw the index of the 3-th dimension of the xD LID
  - param il the index of the 4-th dimension of the xD LID

- double setLampOnOffState(double idx, double setOn);

  sets the on/off state of the idx-th lamp to setOn, i.e when setOn is true the lamp is switched on and when it is false, the lamp is switched off
  - return true if successful
  - param idx index of lamp ranging from 0 to getNumLamps()
  - param setOn the lamp on/off state

6.2. Vehicle functions

These functions allow to get the state of the vehicle in LucidDrive. In the automatic mode you have access to the drive curve. In the manual mode which is available in LucidDrive Manual and LucidDrive ServerClient only, you have access to the steering wheel and gas and brake pedals of an external input device. You have to set the vehicle axis system and the speed (this is needed to show the speed in LucidDrive).

The following functions are listed in alphabetical order.

- double getBrakePedal();

  returns the brake pedal from an input pedal. It is in the range [0,1] with 0 being not pressed and 1 being fully pressed
  - return brake pedal

- double getCurrentTime();

  returns current time in the simulation
  - return current time in seconds

- double getCurvatureAtTime(double t, double out_curvature);

  returns the curvature of the drive curve at a given time
  - return true if successful
  - param t time in seconds
  - param out_curvature Gaussian curvature of the drive curve

- double getDriverRotAngleAtDistance(double t, double ds, double out_angle);

  returns the rotation angle of the head of a driver when she would look at a point on the drive curve at a given distance
  - return true if successful
  - param t time in seconds
  - param ds distance along the drive curve
  - param out_angle angle of head rotation

- double getFrameTime();

  returns time from last frame up to now
  - return frame time in seconds

- double getGasPedal();

  returns the gas pedal from an input pedal. It is in the range [0,1] with 0 being not pressed and 1 being fully pressed
  - return gas pedal

- double getMotorcycleRollAngleAtTime(double t, double out_rollangle);

  returns rollangle of the vehicle assuming it is a bicycle and travels around a curve with curvature and speed given by the functions above
  - return true if successful
  - param t time in seconds
  - param out_rollangle roll angle of the bicycle

- double getPosAndDirAtDistanceAtTime(double t, double ds, double out_time, object out_pos, object out_dir);

  returns the position and direction of the car as it travels along the drive curve at a given time and a given distance from the car position
  - return true if successful
  - param t time in seconds
  - param ds distance along the drive curve
  - param out_time time the point at the given distance will be reached
  - param out_pos position on the drive curve
  - param out_dir tangent on the drive curve

- double getPosAndDirAtTime(double t, object out_pos, object out_dir);

  returns the position and direction of the car as it travels along the drive curve at a given time
  - return true if successful
  - param t time in seconds
  - param out_pos position on the drive curve
  - param out_dir tangent on the drive curve

- double getSpeedAtTime(double t, double out_speed);

  returns the speed of the car at a given time
  - return true if successful
  - param t time in seconds
  - param out_speed speed of the car

- double getSpeedInKmh();

  returns the speed of the vehicle in kilometer per hour
  - return the speed of the vehicle in kilometer per hour

- double getSpeedInMps();

  returns the speed of the vehicle in meter per second
  - return the speed of the vehicle in meter per second

- double getSteeringAngle();

  returns the steering angle from an input wheel. It is in the range [-1,1] with 0 being the 'straight ahead' position
  - return steering angle

- double getTotalLengthOfDriveCurve();

  returns the total length of the drive curve
  - return length of the drive curve

- double getTotalTimeOfDriveCurve();

  returns the total time of traveling the whole drive curve
  - return total time

- double getUAtDistance(double s);
  double getTimeAtDistance(double s);
  double getDistanceAtTime(double s);

  returns parameter of the drive curve at a given distance
  - return the u parameter of the drive curve at a given time
  - param distance in meter

- double getUAtTime(double t);

  returns parameter of the drive curve at a given time
  - return the u parameter of the drive curve at a given time
  - param t time in seconds

- object getVehicleAxis();

  returns the axis system of the vehicle
  - return the axis system of the vehicle

- double intersectRoad(object pos, object isec);

  shoots a ray into the scene from pos in the direction (0,0,-1), and returns the point of the first intersection
  - return true if scene was intersected
  - param pos origin of ray
  - param isec returned intersection point

- double manualDriving();

  returns true if the manual driving mode is on
  - return true if manual driving mode is active, and false if automatic mode if is active

- double setDriverRollInDeg(double deg);

  sets the driver roll angle in degree
  - return true if successful
  - param deg roll angle in degree

- double setDriverRollInRad(double radians);

  sets the driver roll angle in radians
  - return true if successful
  - param radians roll angle in radians

- double setSpeedInKmh(double speed);

  sets the vehicle speed in kilometer per hour
  - return true if successful
  - param speed new speed in kilometer per hour

- double setSpeedInMps(double speed);

  sets the vehicle speed in meter per second
  - return true if successful
  - param speed new speed in meter per second

- double setVehicleAxis(object axis);

  sets the vehicle axis
  - return true if successful
  - param axis new axis to be set

- double setVehicleRollInDeg(double deg);

  sets the vehicle roll angle in degrees
  - return true if successful
  - param deg roll angle in degrees

- double setVehicleRollInRad(double radians);

  sets the vehicle roll angle in radians
  - return true if successful
  - param radians roll angle in radians

Chapter 7. Plug-ins provided in LucidDrive

The main agenda of the use of plug-ins in *LucidDrive* is a modular concept. It is not meant to have one plug-in for every single task/setup (and thereby having a lot of quite similar plug-ins), but to have several smaller ones, each responsible for a part of the whole problem. To ensure that every single plug-in can be activated or deactivated via the plug-in menu independently, furthermore the single plug-ins communicate among themselves, more precise they supply and retrieve the data among each other (the chosen options and values of the parameters). These points together ensure the modularity of the plug-in-system in LucidDrive.

7.1. AFS Scripts

Below some of the provided plug-ins are listed in alphabetical order and shortly explained. They have to be understood as examples, not to adopt them 100% but to give a first sketch how to program and implement the wanted measurements. The single plug-ins can be modified, upgraded and replaced to fit the customers own agenda.

- Some points concerning AFS in general

  When using an AFS script, all headlamps are involved until they are divided in the script into single groups.

  AFS look ahead entry/exit: This determines the point the script tries to illuminate. The script always scans the road in advance (following the driving curve) and points the light onto the look ahead coordinates (of course with constrictions due to the maximal possible swivel angle). It does not e.g. consider the magnitude of a curvature, so the light may illuminate the roadside if there is a sharp turn to come.

  Four of the scripts (having "Demo" in their respective file names) are new demonstrations how to build up an AFS for different types of lamps (high beam, low beam) and purposes, they can be used independently and thus together too.

- afs3DLidScript.do

  This is a more complex version of the usual AFS system. The main difference is that this script contains standard LID in each layer, but having a third coordinate, thus building up a three-dimensional LID (roughly there is a analogy to the volume sensor in LucidShape). Beside u and v the third coordinate is e.g. the curvature of the road, a roll angle etc., allowing LucidDrive to switch between the single 2D-LIDs in dependency on the road. This switching allows a 'better' light on the road but of course it consumes some calculation power. The single LIDs have to be computed and saved separately in order to build up a 3D-LID in LucidShape. The main usage for this type of AFS is to simulate more complex vehicular headlamp systems.

- afsDefault.do

  This is a very basic example of implementing an AFS. It only swivels the light, i.e. there is only a rotation around the z-axis and no rotation about the y-axis (pitch).

- afsDemoLow.do

  This script allows to swivel a user defined lamp or set only. This is done by the parameters to be entered in the respective dialogs. In this case, if the string for "lowbeam set prefix" is identical

to the string entered in the headlamps dialog, this AFS script only swivels the respective set of lamps (e.g. to keep all other lamps headed straight forward onto the road).

- afsDemoHigh.do

  This script uses a different approach, it controls the left and the right high beam part independently. If e.g. an opposing car is approaching on a straight road, the right high beam is unaltered. The left high beam starts to swivel to the left until it reaches a limit, then it will be turned off. After the car has driven past, the left high beam is activated again. The main advantage is not to resign to the right high beam during the passing.

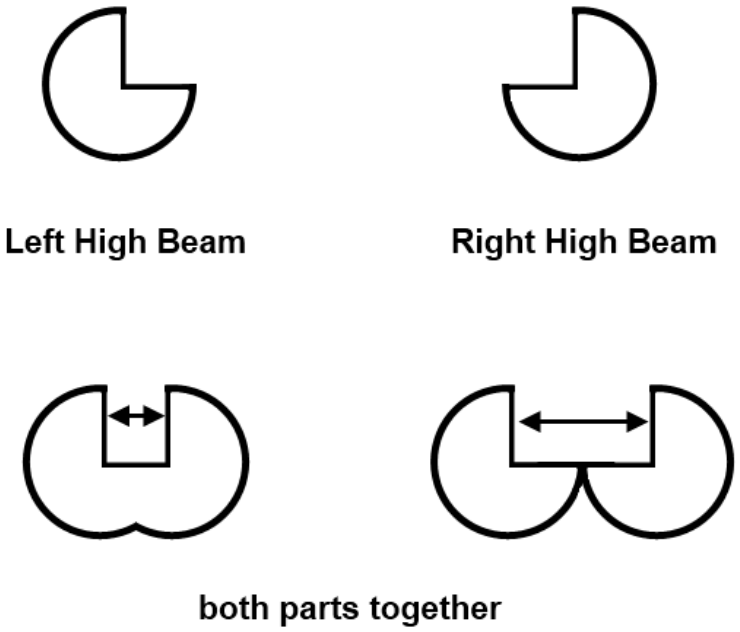


Figure 7.1: A sketch of the concept of this AFS script

A minor disadvantage is the fact that due to the parallax effect it is not possible to reduce the gap between the two parts completely to zero.

- afsDemoTilt.do

  The script afsDemoTilt.do shows an example how to implement a tilt (or pitch), i.e. a rotation around the y-axis. This tries to illuminate the road in regard to dips and cambers and preventing the car from glaring the opposing traffic.

- afsDemoCurveLight.do

  This script allows to control a special set of lamps too. Here additional curve light ("cl", either on the left or on the right side of the car) may be activated when approaching a turn, not changing the pointing of all other lamps. The (de-)activation of the respective cl-lamp is determined by the minimum and maximum curve angle respectively.

- afsDynamicPencil.do

This script controls multiple LEDs in a kind of array, optional arranged in groups. Curve light is realised by temporary activated additional lamps at the front side of the car. The only part of the light being swiveled is the asymmetric part (15° finger), the part lying below the cutoff-line is kept straight forward.

- afsDynamicSpot.do

  Implements a kind of marker, mainly used in static situations e.g. to test a lamp or a texture.

- afsEquimpmentSpot.do

  This script is kind of a tracking system, it lights up certain points along the track. The coordinates to point at are specified via the upper three coordinates (spot target x,y,z look ahead), thus not being restricted to a target directly on the road. Using e.g. the spot target distance values (50,100), every 100m along the driving curve a separate beam/spot points to the coordinates specified, following it 50m before and 50m after passing this point. (Not really used as a usual AFS, but e.g. to route working headlights.)

- afsHighDim.do

  This shows another possibility to avoid glaring other road users. Here the light is not bent away but dimmed down to e.g. 10% of the full intensity.

- afsSkeleton.do

  This script shows a basic setup how to exchange data with Matlab, e.g. to have the AFS in LucidDrive being controlled by an external script.

7.2. Auxiliary Scripts

Additionally there are some auxiliary scripts provided by LucidDrive or LucidDriveManual.

- Audio.do

  LucidDriveManual is not restricted on graphic presentation, this script allows additional sound (only available in the manual driving mode). It is grabbed from a real car and changing in real-time during a running simulation in dependency on the state of the motor engine. The script can downloaded from our website at http://www.brandenburg-gmbh.de/products/lucidshape/downloads/tools/. Download and unpack it into your LucidDrive folder. When activating the script the executable file ("LucidAudioServer.exe") and the directory containing the audio files name have to selected. Press the "..." button and select one of the .wav files in "..LucidDriveManual\LucidDriveAudioPackage\soundBankExample\", the script will automatically load all available audio files in this directory.

- Logger.do

  This is a tool which is not working by itself (it is working, but the output is kind of dummy information how it could work). The main idea is to use the signal-slot mechanism to gather data e.g. from an AFS script. This script also can be coupled with a trigger point to only save data on a specific part along the track.

- Speedometer.do

Additional to the text only display in the status bar, the current speed can be shown graphically to enhance the realistic view during the running simulation. The rpm (revolutions per minute) can only be activated in the manual driving mode, in the automatic mode it is not shown. The size and the position of both graphics can be changed independently. The respective first parameter determines the radius of the display (i.e. the arc), the second is the x-coordinate (x=o is the centre of the car), the third is the y-coordinate with y=o being the centre of the driving view.

Chapter 8. Physics on Geometry, BRDF, Monitor Luminance, etc.

In this new chapter some basics about the physics and made approaches are described, as well as the used angles, some reflection and scattering models etc.

8.1. Geometry

The following sketch visualizes the used geometry, this is intended to show the parameters and achieve geometrical motivated approaches for some different BRDF models. The light comes from the left, intersects with the road surface (middle) and is detected by the observer (right).

$h_{i,o}$ are the heights of the light-source and the observer, $z_{i,o}$ are the horizontal distances (along the road surface). **N** is the normal of the road surface at the intersection point, **I** and **O** are the vectors pointing towards the light-source and the observer, respectively; $\vartheta_{i,o}$ are the respective included angles.

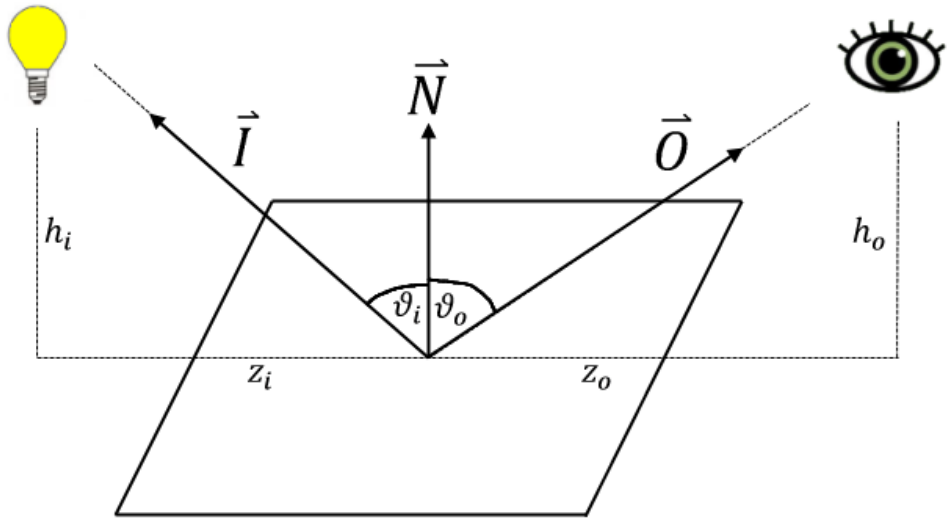


Figure 8.1: A sketch of the scattering process

$$
\begin{aligned}
\text{BSDF}: \quad & \frac{dL_o}{dE_i} \rightarrow f_r(\vec{I},\vec{O}) = f_r(\vartheta_i,\vartheta_o) \\
\text{default BSDF}: \quad & f_r(\vec{I},\vec{O}) = \frac{R}{\pi}\frac{1}{<\vec{I},\vec{N}>} = \frac{R}{\pi}\frac{1}{\cos\vartheta_i} \\
\text{Lommel}-\text{Seeliger}: \quad & f_r(\vec{I},\vec{O}) = \frac{R}{\pi}\frac{1}{<\vec{I},\vec{N}>+<\vec{O},\vec{N}>} = \frac{R}{\pi}\frac{1}{\cos\vartheta_i+\cos\vartheta_o}
\end{aligned}
\tag{8.1}
$$

More precise the differential ratio in the first equation has to be determined pixel by pixel for each point on the road surface.

8.2. BRDF

In general a BRDF determines the ratio of the incoming luminance (radiance) and the outgoing illuminance (irradiance). The first approach is to assume an isotropic behaviour of the scattering process about ψ, the angle inside the road surface plane. This leads to the BRDF model of Lommel-Seeliger.

The next approach is to assume that z_0 is approx. 1 and $z_0 \gg h_0$, so for the limes of $\vartheta_0 \to 90°$ we can neglect this summand in the fraction and obtain the BRDF used as by default in LucidDrive.

8.2.1. Hoffmann & Nakata

Nakata and Hoffmann are two empirical models which are characterised in von Hoffmanns dissertation (see also [2]). Both of them already contain the reflection rate of the measured street, therefore they appear darker then the default BRDF does.

The default BRDF model for rough surfaces in LucidDrive is: $f(\vartheta_i) = R/(\pi \cdot \cos\vartheta_i)$.

The motivation behind this choice is to take the most simple function which looks realistic and is not dependent to the view direction. The models of Hoffmann and Nakata are measurement models used for backward scattering. Both have the form : f(z) = A·z+B (where z is the distance from the illuminated area to the light source in horizontal direction).

Hoffmann: A = 0.02279; B = 0.029722
Nakata: A = 0.07578; B = 0.0331

So both of them are linear fittings to the measurement results of backward scattering on real streets, thus they already contain the reflexivity of the road surface. For the Lambert model in contrary this is not taken into account. Here the reflexivity is included via the parameter "reflection index" in the appearance dialog (in LucidDrive by default $1/\pi$).

If you achieve good results using the Nakata BRDF model, the reflexivity of the street in the luminance measurements are similar to Nakata. To obtain the reflexivity for our default BRDF, one can do it by equating coefficients:

$$\frac{R}{\pi \cdot \cos\vartheta_i} = A \cdot z + B \qquad \frac{z}{h} = \tan\vartheta_i = \frac{\sin\vartheta_i}{\cos\vartheta_i}$$

$$\begin{aligned} \Rightarrow R &= \pi \cdot \cos\vartheta_i (A \cdot z + B) \\ &= \pi \cdot \cos\vartheta_i (A \frac{h \sin\vartheta_i}{\cos\vartheta_i} + B) \\ &= \pi A h \sin\vartheta + \pi B \cos\vartheta_i \\ \lim_{\vartheta_i \to 90°} \Rightarrow R &= \pi h A \sim 0.15 \quad (\text{using } A = 0.07578 \wedge B = 0.0331) \end{aligned} \tag{8.2}$$

When having a street with a "reflection index" of 0.15 for road surface and side green, Nakata fits quite good to the default BRDF model.

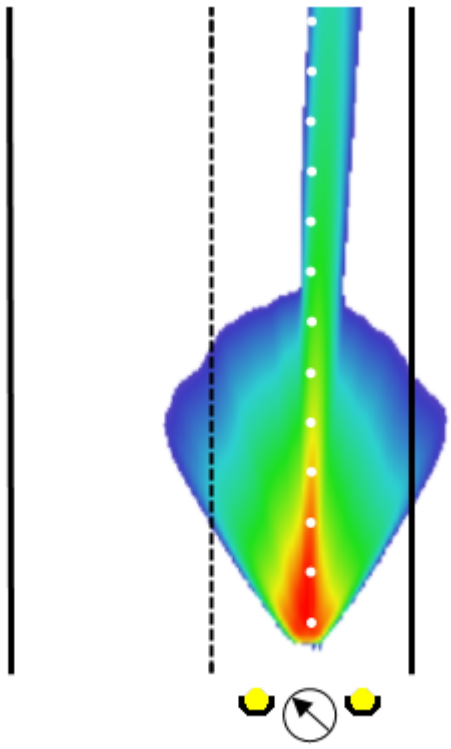


Figure 8.2: A sketch of a view along the street

The figure right above shows the measurement method done by Hoffmann and Nakata (independently of each other). Both of them set a car on the road with two lamps attached and a sensor between them. Then they have measured the light being scattered back from the measuring points on the surface (white dots; the LID is oversimplified just to illuminate the measuring zone on the road). This method of measurement leads both to a good simplification but to a large restriction of the validity area too. This type of BRDF does only work when observing the scattering process from a position very close to the driver view.

The corresponding parameters and used values in von Hoffmanns dissertation (in chapter 6) are: $h_o = h_S = 0.65m$ and $h_i = h_A = 1.13m$.

8.2.2. Lommel-Seeliger

The Lommel-Seeliger is a BRDF mostly used in astronomical tasks for description of planet surfaces. It has the form $f(\vartheta_i, \vartheta_o) = R/[\pi \cdot (\cos\vartheta_i + \cos\vartheta_o)]$.

Please refer also to http://orca.phys.uvic.ca/~tatum/plphot/plphoto1.pdf.

For the intensity mapping at first the normalised luminance is calculated, here the maximal luminance L_{max} is involved: $L_n = L/L_{max}$. The light intensity function for Munsell is an approximation to the light intensity component of the *Munsell Color Order System*. The following listing contains a corresponding source code:

```
float munsell (float Ln) {
  const float c1 =  0.11913;
  const float c2 = -0.22533;
  const float c3 =  2.3352;
  const float c4 = -2.0484;
  const float c5 =  0.8194;
  const float d2 = -0.45066;
  const float d3 =  4.6704;
  const float d4 = -4.0968;
  const float d5 =  1.6388;

  float r = pow(Ln, 0.45);
  float f = r * (c1 + r * (c2 + r *(c3 + r * (c4 + r * c5))));
  float df = c1 + r * (d2 + r * (d3 + r * (d4 + r * d5)));
  r = r - (f - Ln) / df;
  return r;}
```

8.3. Illumination & Luminance Parameters

The following table lists some of the parameters involved with a short description (in alphabetical order). The nomenclature of course is not stringent, but tries to use descriptive names/shortcuts.

$A = A(P)$	Direction from P to the eye
Amb	Ambient intensity setting in LucidDrive
$B = BM(LN)$	Value of the brightness after brightness mapping (BM) from [0,1] $\rightarrow$ [0,1]
BRDF(X,Y)	BRDF of the plane, when light comes from -X and goes to Y
$E = I \cdot \langle N,L_i \rangle \cdot r^{-2}$	Sum for all light sources
$I = I(L_i,P)$	Intensity of the i-th PLS in direction $-L_i(P)$
$L = L(P,X) = E \cdot R \cdot BRDF(L_i,A)$	Luminance at P in direction of X
$L_i = L_i(P)$	Direction from P to the i-th point light source (PLS)
L_{max}	Maximal luminance setting in LucidDrive
$L \geq L_{max} \rightarrow LN = 1$	
$L < L_{max} \rightarrow LN = L/L_{max}$	
LN	Normalised luminance
LRGB	Color of the light-source
$N = N(P)$	Normal of the surface at P
$P = P(x,y)$	Next intersection of the ray from the eye/camera through the pixel (x,y) with the scene
R	reflection rate of the surface
$RGB = RGB(B,SRGB)$	Output value of the RGB
$RGB = SRGB \cdot B \cdot (LRGB + Amb)$	
$r2 = r2(P,L_i)$	Square of the distance from P to i-th PLS
$SRGB = SRGB(x,y)$	Value of the RGB on the surface at pixel (x,y)
$\langle X,Y \rangle$	Inner product of X,Y; if both are vectors (i.e. directions) with $\|X\|=1$ $\rightarrow \langle X,Y \rangle = \cos(\angle(X,Y))$

What then, after all the calculations have been done, happens while displaying these RGB values on the screen (gamma curve), is beyond our control because this strongly depends on the respective monitor. But this should be adjustable using a special brightness mapping. Changeable parameters are BRDF, reflection rate, brightness mapping and L_{max}.

8.4. Monitor and Luminance

Most Monitor Calibration Tools adjust the color appearance, but not the luminance of the display. If you want to get a decent color appearance, just use any of the systems on the market (e.g. 'EyeOne' or 'ColorSpyder').

In order to match the luminance levels to the actual monitor luminance you should find out what luminance your monitor produces with all pixels set to white (it should be about 80-130 cd/m^2 with a brightness setting at 30% on recent monitors).

Then you can insert this luminance as the maximum value for the luminance range in the RoadProperties dialog in LucidDrive (see the picture below). The luminance response of a monitor for equal RGB values X from 0 to 1 roughly follows a X^2 curve. The brightness mapping is the inverse of this curve and is in this case a Square Root function.

You can switch to such a setting with the brightness mapping function "Custom" in the RoadProperties dialog (see picture below). Then the actual monitor output matches quite well the calculated luminance values from LucidDrive. You can also go further and measure the response curve of the monitor and we could implement a special brightness mapping curve from that.

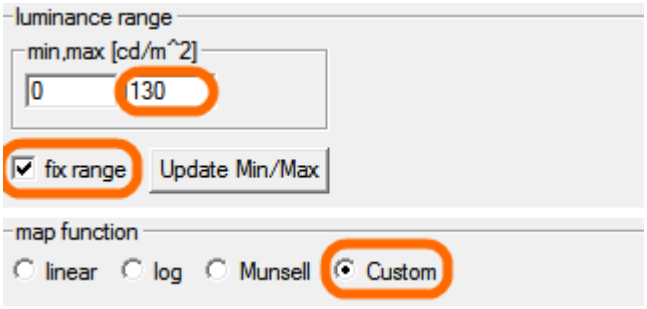


Figure 8.3: The Dialog parts with the Settings explained above

Chapter 9. LucidDrive Keyboard Shortcuts

9.1. LucidDrive Shortcuts

s	Resets the driving simulation and go back to the start.
g	Starts the driving simulation.
h	Halts the running driving simulation.
n	Goes to the next time step in the simulation.
b	Goes to the previous time-step in the simulation.
m	Toggles between manual/automatic driving.
d	Switches to driver's view.
u	Switches to the 20m view.
t	Switches to the bird's eye view.
p	Switches to the pedestrian's view.
y	Switches to the drive-by view.
r	Rotates the view to the right.
l	Rotates the view to the left.
i	Rolls the view to the left.
j	Rolls the view to the right.
o	Resets the rotation of the view.
0..9	Switches exclusively between the lamp sets.
+	Switches to the next lamp set.
-	Switches to the previous lamp set.
F12	Toggles the full screen viewing mode on/off.

9.2. LucidDriveManual and LucidDriveServerClient Extra Shortcuts

e	Exit LucidDrive and all connected clients.
a	Reset LucidDrive and all clients.
arrow keys	Drive manually, up/down is gas/brake, left/right is steering wheel input (only if no external steering wheel is connected).

Bibliography

[1] TECHNICAL COMMITTEE ISO/TC29/SC9. *Road vehicles – Vehicle dynamics and road-holding ability – Vocabulary*.

[2] ALEXANDER VON HOFFMANN. *Lichttechnische Anforderungen an adaptive Kraftfahrzeug für trockene und nasse Fahrbahnen, Dissertation TU Ilmenau*.